ANSI/ISO/IEC 8211-1994

American National Standard

for Information Technology –

Specification for a Data Descriptive File for Information Interchange

ANSI American National Standards Institute
11 West 42nd Street
New York, New York
10036

American National Standard

Approval of an American National Standard requires review by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

Published by

American National Standards Institute
11 West 42nd Street, New York, New York 10036

Printed in the United States of America

ANSI/ISO/IEC 8211-1994
revision and redesignation of
ANSI/ISO 8211-1985 (R1992)

American National Standard
for Information Technology –

Specification for a Data Descriptive File for Information Interchange

Secretariat
Information Technology Industry Council

Approved February 20, 1995
American National Standards Institute, Inc.

Contents

List of figures

List of tables

Foreword (This foreword is not part of American National Standard ANSI/ISO/IEC 8211-1995. This document is identical to ISO/IEC 8211-1994 and the following seven paragraphs are the original foreword as it appeared in that document.)

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to the national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 8211 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 21, *Open sytems interconnection, data management and open distributed processing.*

This second edition cancels and replaces the first edition (ISO 8211: 1985), which has been technically revised.

The substantive changes made to produce this edition of ISO/IEC 8211 are the following additions:

1. Binary forms for numeric values.
2. Binary leaders and directories.
3. Support for ISO/IEC 10646.
4. Definition of FTAM unstructured and structured document types.
5. Concatenated regular structures.
6. Recursive tree description.
7. A human-readable, alternate form of data field description.

The second edition is backwards compatible with the first edition.

Annex A forms an integral part of this International Standard. Annexes B to H are for information only.

Requests for interpretation, suggestions for improvement or addenda, or defect reports are welcome. They should be sent to the X3 Secretariat, Information Technology Industry Council, 1250 Eye Street, NW, Suite 200, Washington, DC 20005.

This standard was processed and approved for submittal to ANSI by Accredited Standards Committee on Information Technology, X3. Committee approval of this standard does not necessarily imply that all committee members voted for its approval. At the time it approved this standard, the X3 Committee had the following members:

James D. Converse, Chair
Donald C. Loughry, Vice-Chair
Joanne Flanagan, Secretary

Organization Represented	*Name of Representative*
American Nuclear Society	Geraldine C. Main
	Sally Hartzell (Alt.)
AMP, Inc.	Edward Kelly
	Charles Brill (Alt.)
Apple Computer, Inc.	David K. Michael
AT&T Global Information Systems	Robert K. Kramer
	Thomas F. Frost (Alt.)
Bull HN Information Systems, Inc.	William M. George Jr.
Compaq Computers	Mitesh Patel
Digital Equipment Corporation	Scott K. Jameson
	Richard Hovey (Alt.)
Eastman Kodak Company	James D. Converse
	Michael Nier (Alt.)
Guide International, Inc.	Tony Gualtieri
	Harold Kuneke (Alt.)
Hewlett-Packard	Donald C. Loughry
	Karen Higginbottom (Alt.)
Hitachi America, Ltd.	John Neumann
	Kei Yamashita (Alt.)
Hughes Aircraft Company	Harold Zebrack
IBM Corporation	Joel Urman
	Mary Anne Lawler (Alt.)
Institute for Certification of Computer Professionals (ICCP)	Kenneth Zemrowski
National Communications Systems	Dennis Bodson
	Granger Kelley (Alt.)
National Institute of Standards and Technology	Michael D. Hogan
	James H. Burrows (Alt.)
Neville & Associates	Carlton Neville
Northern Telecom, Inc.	Mel Woinsky
	John Pugh (Alt.)
Recognition Technology Users Association	Herbert P. Schantz
	Gerald Farmer (Alt.)
Share, Inc.	Gary Ainsworth
	David Thewlis (Alt.)
Sony Corporation of America	Michael Deese
Storage Technology Corporation	Joseph S. Zajaczkowski
	Samuel D. Cheatham (Alt.)
Sun Microsystems, Inc.	Gary S. Robinson
Sybase, Inc.	Donald R. Deutsch
Texas Instruments, Inc.	Clyde Camp
	Fritz Whittington (Alt.)
3M Company	Eddie T. Morioka
	Paul D. Jahnke (Alt.)
Unisys Corporation	John L. Hill
	Stephen P. Oksala (Alt.)
U.S. Department of Defense	William Rinehuls
	C. J. Pasquariello (Alt.)
U.S. Department of Energy	Alton Cox
U.S. General Services Administration	Patrick Plunkett
	Douglas K. Arai (Alt.)
Wintergreen Information Services	Jack L. Wheeler
Xerox Corporation	Roy Pierce
	William Ted Smith (Alt.)

Introduction

This International Standard has been produced in response to an identified need for a mechanism to allow data structures to be easily moved from one computer system to another, independent of architecture Data structures required to be interchanged can vary significantly in complexity and size, and a common method to accomplish these interchanges is desirable It is also desirable that any medium such as a communication line, a magnetic tape, a disk pack, a flexible disk etc , should be able to be used for the physical interchange, and that all information necessary to successfully recreate the structure in the target system should be contained within the information transported on the medium

To meet these needs this International Standard specifies medium-independent and system-independent file and data record formats for the interchange of information between computer systems This International Standard is intended for use with physical recorded media as well as with communications media The contents in the user data structure can be of any internationally recognized character set and coding and are interchanged in a transparent fashion The intermediate structure through which the information passes is designed for interchange purposes It can also be used for some forms of general processing and is amenable to direct access methods on high volume, direct access interchange media

This International Standard is a concrete transfer syntax and encoding standard and provides a tool for the description of files containing user data but does not specify the content or order of user data fields or user data records It does specify a comprehensive generic form for such records and fields which can accommodate a wide variety of user needs for both simple and complex user data An application must design its own instance of a conforming interchange file and all conforming files, both data and data description, will be processable to the field or subfield level by the same software A user must, of course, complete the interface to their own application system

The approach used is to define an interchange format into which most information structures and their content can be transformed without loss of information, and from which the original structure and content can be retrieved The interchange format is suitable both for recording on physical media and transport through a communication system

The data structures supported by the interchange format are elementary data, vectors, arrays and hierarchies The file structures that can be transformed into the interchange format include sequential, hierarchical and relational Network structures are not directly supported and additional pre-processing and post-processing are necessary in this case to preserve logical linkages.

This International Standard is media independent It assumes, at a minimum, that the supporting transport system can process fixed length octet strings It requires a computer processing capability to map the user file or database data to the interchange file This mapping function has to provide the necessary data and structure conversions The parameters required to define the selection and conversion of these data items and structures into the formats specified by this International Standard are outside the scope of the standard This International Standard requires the use of a basic character set based on ISO 8859-1 and ISO/IEC 6429 in control fields and permits the use of additional character sets in user data fields This International Standard provides for three interchange levels from which the users may choose based on the complexity of their data structures The first interchange level supports multiple fields containing simple, unstructured character strings The second level supports the first level and multiple fields containing structured user data comprising a variety of data types. The third level supports the second level and hierarchical data structures

The experience of implementing ISO/IEC 8211 for a variety of applications revealed the need for the changes introduced in this version Many of the changes give ISO/IEC 8211 increased versatility and more effective interchange capabilities Many other changes were made to improve clarity and user acceptance. Technical changes in the standardized interchange supporting this International Standard and changes in the organizational responsibility for this International Standard have led to other extensions This version provides the user with an improved interchange tool in keeping with the user's increasing needs and well integrated into the OSI environment

The retrieval of archived files may require the use of computer systems which are different from the original archiving systems The operational problems are identical to those involved in the transport of files between computers at different sites and this International Standard provides a facility for this application

ISO/IEC 8211 is based on ISO 2709 having the same record structure but different data description components ISO 2709 based systems for file transfer and random file access had been in use since at least 1970 and their use is now extensive The nomenclature of ISO/IEC 8211 conforms to its predecessor, ISO 2709 Its hierarchical, logical data constructs are files, variable-length records, variable-length fields and subfields In several programming languages, the equivalent of an ISO/IEC 8211 field is a record and specific applications will transport their records as ISO/IEC 8211 fields with related records aggregated into ISO/IEC 8211 records

Organization of the Standard

The contents of this standard are organized as follows

1) Clause 5 describes the specifications of the contents of leaders, directories and field areas common to all logical records and necessary to the import of logical records and complete fields
2) Clause 6 describes the specification of the data description necessary to import user data at the subfield level Subclause 6.1 specifies further subfields in the DDR leader which contain information pertinent to data description
3) Clause 7 describes the use of extended character sets
4) Annex A describes the FTAM registrations
5) Annex B describes a methodology for specifying ISO/IEC 8211 file design and data descriptions
6) Annexes C through H provide tutorial information on the methodology The reader may wish to read Annex C prior to studying Clause 5 and Annex D prior to studying Clause 6

American National Standard
for Information Technology –

Specification for a Data Descriptive File for Information Interchange

1 Scope

This International Standard specifies an interchange format to facilitate the moving of files or parts of files containing data records between computer systems The interchange format is not intended as a record format for the indigenous files of any specific system but may be used for this purpose The standard defines a generalized structure which can be used to transmit, between systems, files or records containing a wide variety of data types and data structures It specifies the means for the description of the contents of data records but does not specify their application semantics although these semantics can be included as a part of the transmission The interchange format may also be used to transport individual records, individual data fields or individual subfields with their description

This International Standard specifies

- media-independent file and data record descriptions for information interchange,
- the description of data elements, vectors, arrays and hierarchies containing character strings, bit strings and numeric forms;
- a data descriptive file composed of a data descriptive record and companion data records that enable interchange to occur with minimal specific external description,
- the data descriptive record that describes the characteristics of each data field within the companion data records,
- three levels of complexity of file and record structure;
- FTAM unstructured and structured document types

2 Normative references

The following standards contain provisions which, through reference in the text, constitute provisions of this International Standard At the time of its publication, the editions indicated were valid All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below Members of IEC and ISO maintain registers of currently valid International Standards

ISO/IEC 646 1991, *Information technology - ISO 7-bit coded character set for information interchange*

ISO 2022 1986, *Information processing - ISO 7-bit and 8-bit coded character sets - Code extension techniques*

ISO 6093 1985, *Information processing - Representation of numerical values in character strings for information interchange*

ISO/IEC 6429 1992, *Information technology - Control functions for coded character sets*

ISO 8571-1 1988, *Information processing systems - Open Systems Interconnection - File Transfer, Access and Management - Part 1 General Introduction*

ISO/IEC 8824 1990, *Information technology - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN 1)*

ISO 8859-1 1987, *Information processing - 8-bit single-byte coded graphic character sets - Part 1 Latin alphabet No. 1*

ISO/IEC 9834-2—1993, *Information technology - Open Systems Interconnection - Procedures for operation of OSI Registration Authorities Part 2. Registration procedures for OSI document types*

ISO/IEC 10646-1 1993, *Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1. Architecture and Basic Mulitlingual Plane*

IEC 559 1989, *Binary floating point arithmetic for microprocessor systems* (also ANSI/IEEE 754 1985(R1991))

The following document is also relevant to this International Standard

ISO International register of coded character sets to be used with escape sequences

3 Conformance

Interchange files conform to this International Standard when all of the data descriptive records and data records conform to the specifications of this International Standard A statement of conformance shall specify the version number and interchange level to which the contents of files conform

This International Standard does not specify requirements for processing and implementation, therefore such processing and implementation cannot itself conform to this International Standard

4 Definitions and abbreviations

4.1 Definitions

For the purposes of this International Standard the following definitions apply

4.1.1 alphanumeric character A character occurring in columns 2 to 7 inclusive (except position (7/15)) of the Basic Character Set or the corresponding characters of ISO/IEC 10646 Table 1 - Row 00 Basic Latin These characters correspond to those of ISO/IEC 646 IRV

NOTE 1 - The characters specified in this International Standard are represented by their position (column/row) in the coded character set table or by their acronym or name, e g , ESC, SPACE and DIGIT ZERO or "0" Alphanumeric characters which are explicitly specified in a control field are enclosed in double quotes, e g , "1" The special ISO/IEC 8211 delimiters unit terminator(1/15) and field terminator(1/14) are represented by UT and FT respectively

4.1.2 array A data structure of two or more dimensions

4.1.3 array descriptor A Cartesian label or numeric array descriptor which provides a description of the dimension and extents of an array

4.1.4 base address of data A data element the value of which is equal to the octet count up to but not including the first octet of the first data field following the field terminator of the directory, where the specified origin (0) is the first octet of the leader

4.1.5 basic character set; BCS A character set comprising a) ISO 8859-1 including SPACE as the G0 and G1 sets and b) ISO/IEC 6429 as the C0 and C1 sets

NOTE 2 - This set is compatible with row 00 of the ISO/IEC 10646 Basic Multilingual Plane and should greatly reduce any dependence on national variant character sets However there currently are differences in the allowed uses of the C1 set between ISO/IEC 6429 and ISO/IEC 10646

4.1.6 binary form A binary form is a sequence of one or more octets having a specified format and meaning The octets of a binary form may be recorded in order of decreasing significance with the most significant octet first (MSOF) or the reverse order, least significant octet first (LSOF) The bits within an octet are always recorded in order of decreasing significance

4.1.7 binary tree A rooted tree in which each node has zero subtrees or, at most, two subtrees known as the left subtree and the right subtree In the absence of either the left or right subtree, the remaining subtree retains its identity as a left or right subtree See annex F.

4.1.8 bit field A data field comprising only binary digits When it is necessary to complete an octet, it is filled on the right with binary zeros See also **character mode bit string**.

4.1.9 Cartesian label An array of identifiers formed by the Cartesian product of the elements of two (or more) vector labels The array elements have the same order as the elements of the direct product such that if a and b are the vector labels, where a = (a(1) ..., a(*n*)) and b = (b(1) ..., b(*m*)), then the Cartesian label, a*b = (a(1)b(1), a(1)b(2), ..., a(1)b(*m*), ..., a(*n*)b(*m*)), where a(*i*)b(*j*) is a concatenation of a(*i*) and b(*j*) which forms an identifier of the *j,i*-th element of a corresponding data array (i e., the "a"s are the row labels and the "b"s are the column labels) The Cartesian label is the most general form of a label and contains a vector label and a single label as special cases. The expansion of a higher order Cartesian label is from the left, i e., a*b*c = (a*b)*c, that is, a*b is expanded first

4.1.10 character mode bit string A sequence of alphanumeric characters, i e "0" or "1", that represents a string of binary digits See also **bit field**

4.1.11 compound data field A field comprising one or more subfields each containing an elementary data element.

4.1.12 corresponding binary tree A binary tree which represents the structure of an ordered, rooted tree The first (leftmost) offspring of a node in an ordered rooted tree becomes the root of the left subtree of the binary tree and the siblings of the node are connected through a series of right-hand links The preorder traversal sequence of an ordered, rooted tree and its corresponding binary tree are identical

4.1.13 data descriptive field: A tagged field, residing in the DDR, containing the ISO/IEC 8211 data description for all data fields in the file associated with the same tag

4.1.14 data descriptive file; DDF A file containing a data descriptive record and its companion data records.

4.1.15 data descriptive record; DDR A logical record that contains the control parameters and data definitions necessary to interpret its companion data records The data descriptive record is the first logical record of a file

4.1.16 data field: A tagged field which contains user data and is located in a DR

4.1.17 data record; DR A logical record containing user data.

4.1.18 delimited structure A data structure composed of a collection of data elements that are separated by delimiters

4.1.19 delimiter A single character that separates data elements and data subfields. See 6 4 1, Table 1 for the use of delimiters

4.1.20 directory A field containing an array of identifiers and references to corresponding items of data. It is terminated by a field terminator

4.1.21 directory entry A fixed-length field within the directory that contains information about the field tag, the length and the location of a specific field within a given record

4.1.22 elementary Having the property of being indivisible without loss of meaning.

4.1.23 entry map A field in the leader that is used to indicate the structure of the entries in the directory.

4.1.24 escape character; ESC A control character which is used for code extension purposes Ii causes the meaning of a limited number of bit combinations following it to be changed The use of ESC is specified in ISO 2022.

4.1.25 external file title: A string of characters that provides a displayable descriptive title for the interchange file This need not be the same as the host system file name

4.1.26 field: A generic term used to describe a contiguous set of octets containing one or more values The components of a field are referred to generically as subfields See also **subfield** 2)

4.1.27 field tag; tag A character string in a directory entry used to identify a data field or an associated data descriptive field

4.1.28 field terminator; FT A character, (1/14), used to terminate a variable-length field within a record.

4.1.29 file A collection of related records treated as a unit

4.1.30 forest An ordered set of disjoint, ordered, rooted trees

4.1.31 interchange format A format for the exchange, as opposed to the local processing, of data.

4.1.32 interchange level; level The designation of a prescribed subset of the requirements of this International Standard.

4.1.33 leader A fixed-length field that occurs at the beginning of each record and provides parameters for the processing of that record

4.1.34 least significant octet first; LSOF An ordering of octets in which the least significant octet is placed closest to the beginning of a file

4.1.35 location; position The octet count to the position of the first octet of a field Locations in the leader and directory are relative to the first (0) octet of the leader, and the locations of fields are relative to the base address of data

4.1.36 map; to map To establish the correspondence between the elements of two different data structures

4.1.37 media record The physical record which is characteristic of a medium and which is written on the media by a single system level write statement, e g , a block or a sector or a transmission packet (see also **record**)

4.1.38 most significant octet first; MSOF An ordering of octets in which the most significant octet is placed closest to the beginning of a file

4.1.39 n-tuple A vector, each instance of which has n ordered elements

4.1.40 null The non-occurrence of an entity, usually a data element, string or set

4.1.41 numeric array descriptor A sequence of numbers which specifies the number of dimensions and extent of each dimension of an array

4.1.42 ordered, rooted tree; tree; hierarchy A data structure comprising a set of nodes having directed links such that one node, the root node, has no incoming links, the remaining nodes have precisely one incoming link, and the order of multiple outgoing links from a node is significant The nodes are referred to as parent nodes and offspring nodes and link direction is from parent node to offspring node.

4.1.43 position see **location**

4.1.44 preorder traversal sequence A sequence of the nodes of an ordered, rooted tree which is produced by the following recursive algorithm:

a) Enter the tree at the root node
b) Traverse the left-most subtree not previously traversed,
   if no such subtree exists, stop
c) If b) is not possible, return to the node superior to the subtree and go to b)

4.1.45 record; logical record A contiguous string of octets in an interchange file, the first five being the record length (see also **media record**)

4.1.46 record length A data element the value of which is equal to the length in octets of the record

4.1.47 recursive tree A tree in which a node appears in its own subtree

4.1.48 relative position; RP The position of an octet expressed as a decimal integer relative to the beginning of a record, field or subfield The relative position of the first octet is numbered "0 "

4.1.49 signed binary integer A two's complement binary number

4.1.50 subfield 1) A contiguous string of octets in a field whose position, length and data type are described in the field data description, 2) generically, a component of a field

4.1.51 subfield label A character string used in the data description to identify a subfield and its contents

4.1.52 tagged field A contiguous string of octets in a record which is identified by a field tag and whose length and relative position are specified in a directory entry A tagged field may occur in the DDR or in the DRs

4.1.53 tag see field tag

4.1.54 unit terminator; UT A character, (1/15), used to delimit several types of subfields within variable-length fields in both DDR and DR

4.1.55 variable-length field A field, the length of which may vary from occurrence to occurrence

4.1.56 vector A one dimensional data structure each instance of which may have a variable number of elements (see also **n-tuple**)

4.1.57 vector label A vector the elements of which are labels used to identify each element in a vector of data elements A vector label is a special case of a Cartesian label

4.2 Abbreviations

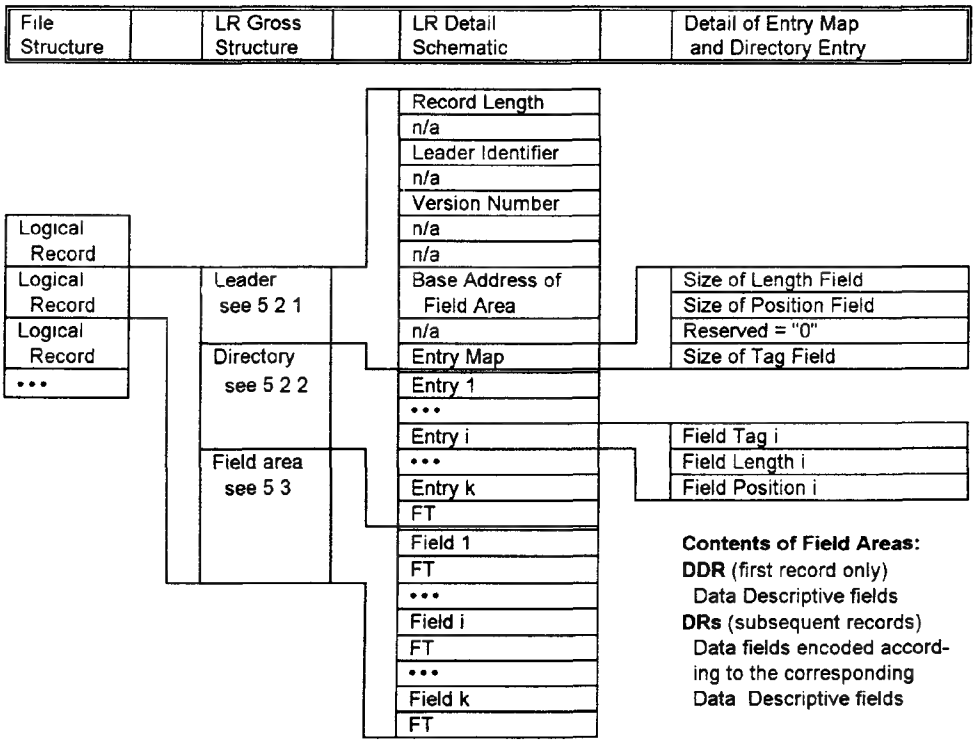
For the purposes of this International Standard the following abbreviations apply

BCS	basic character set
DDF	data descriptive file
DDR	data descriptive record
DFD	data field description, see B.2
DR	data record
ESC	escape character
FT	field terminator
FTAM	file transfer and access management
LR	logical record
LSOF	least significant octet first
MSBF	most significant octet first
RP	relative position
SPACE	SPACE character
TAB	horizontal tabulation character
UT	unit terminator
level	interchange level
tag	field tag
0...n	a tag with zero or more leading ZEROs

5 Interchange file and logical record structure

This clause specifies the general requirements for an interchange file and the requirements which are common to all logical records These requirements specify the structure of logical records external to the fields The requirements for the contents of the fields and the description of the contents of data fields are given in clause 6 See Annex C for an informal description of the constructs of this clause

An introductory schematic for ISO/IEC 8211 files, logical records and their components is shown in Figure 1



Legend.

•••	Omitted entries, fields or records
n/a	This field not used in logical record processing
FT	Field Terminator

Figure 1 - Schematic of ISO/IEC 8211 File and Logical Records

The leaders of all records contain the parameters necessary to read records and disaggregate the directory into its entries Additionally the DDR leader contains a few data descriptive parameters applicable to the entire file

The directories of all records contain the parameters necessary to identify and locate each field in the Field Area

In the first record only, the DDR, the Field Area contains data descriptive fields, each containing the information necessary to decode the user data in the Field Areas of the following DRs.

5.1 File and logical record structure

This International Standard specifies Data Descriptive Files (DDFs), each comprising Logical Records (LRs)

The files may be recorded on a physically dismountable medium or transported through a communications system When recorded on a medium for which an International Standard for volume and file structure exists, each file shall have the required International Standard interchange file labels or headers for the particular medium The file, when recorded on a suitable media volume, may be accessed sequentially or randomly.

An ISO/IEC 8211 interchange file set shall comprise one or more instances of the collection of items depicted in figure 2

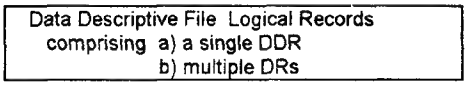


Figure 2 - File Schematic Representation

The DDR is the first logical record of each DDF

5.1.1 Interchange logical records

The logical records specified in this International Standard shall be written, blocked and spanned, across the fixed-length, media physical records (blocks or sectors) or transmission packets without any further record demarcations

5.1.2 Padding of records and media blocks

Any unused portion of a fixed-length, logical record without a leader/directory and any unused portion of a media block or its equivalent shall be padded with the CIRCUMFLEX character, (5/14)

NOTE 3 - This specification gives a satisfactory and detectable end of file condition under all known circumstances The logic for processing the end-of-data and end-of-file condition is discussed in C 1 6

5.2 Logical records - leaders and directories

This subclause contains specifications for logical records and their fields and for the parameters of leaders and directories of logical records which are involved in record processing Both numeric and binary encodings of some leader and directory parameters pertaining to the octet counts are supported and are described in 5.2.1 6 The DDR leader parameters involved with data description are described in 6 1

The logical records shall consist of the areas shown in figure 3

Name of Area	Length
Leader	24
Directory	$k \times p + 1$
Field Area	variable

where k is the number of directory entries and
p is the number of octets in a directory entry of the record.

Figure 3 - Logical Record Schematic

5.2.1 Logical record leader

Each LR Leader shall consist of the fields shown in figure 4 and which are further specified in 5.2 1 1 to 5 2 1 5 and 6.1

The "Use" column in figure 4 designates the use of each field as a record processing control "All" indicates that a meaningful value exists in the field for all records, "DDR" indicates that a meaningful value exists in the field for the DDR only

RP	Len	Use	Field Name	LR Leader Contents DDR	DR	Reference
0	5	All	Record Length	octets	octets	5 2 1 1
5	1	DDR	Interchange Level	"1" | "2" | "3"	SPACE	6 1 1
6	1	All	Leader Identifier	"L"	"D" | "R"	5 2 1 2
7	1	DDR	Inline Code Extension Indicator	character	SPACE	6 1 2
8	1	DDR	Version number	SPACE | "1"	SPACE	5 2 1 3
9	1	DDR	Application Indicator	character	SPACE	6 1 3
10	2	DDR	Field Control Length	digits	SPACEs	6 1 4
12	5	All	Base Address of Field Area	octets	octets	5 2 1 4
17	3	DDR	Extended Character Set Indicator	characters	SPACEs	6 1 5
20	4	All	Entry Map	digits	digits	5.2.1.5

Figure 4 - LR Leader Schematic

5.2.1.1 record length field (LR RP 0-4)

This field shall specify the total length of the LR in octets including the five octets of this field (see 5 2 1 6 for the syntax)

5.2.1.2 leader identifier field (LR RP 6)

This field shall identify the type of logical record and specify the nature of the leader and directories

When all remaining data records in a DDF would have identical leaders and directories, it is permissible to place an "R" in its Leader Identifier field and to omit the leader and directory from subsequent records,

The allowed values and meanings of the leader identifier shall be

L	means that the record is the Data Descriptive Record
D	means that the record is a Data Record and that the next data record has a leader and directory
R	means that the record is a Data Record and that a leader and directory will not be found in any of the subsequent LRs The leader and directory of the current LR shall be applied to each of the subsequent LRs.

5.2.1.3 ISO/IEC 8211 version number (LR RP 8)

In the DDR, this field shall contain the identification of the version of ISO/IEC 8211 to which an interchange file conforms The version number places conditions and constraints on both record and data description processing The allowed values and meanings shall be

Value	Meaning
SPACE	ISO/IEC 8211 - 1985
"1"	this version

This field is unused in the remaining logical records, the DRs, and shall contain SPACE

5.2.1.4 Base address of field area (LR RP 12-16)

This field shall specify the relative position of the first field of the Field Area and its value shall be equal to the combined length in octets of the leader and directory including the field terminator at the end of the directory (see 5 2 1 6)

5.2.1.5 Entry map field (LR RP 20-23)

RP	Name of Subfield	Length	Contents
20	Size of Field Length Field	1	digit
21	Size of Field Position Field	1	digit
22	Reserved for future standardization	1	DIGIT ZERO
23	Size of Field Tag Field	1	digit

Figure 5 - LR Entry Map Schematic

This field specifies the lengths of directory entry subfields in each LR and shall consist of the subfields shown in figure 5 (see 5 2 1 6)

5 2 1 5 1 Size of field length field (LR RP 20)

This subfield shall specify the size in octets of the Field Length field of the directory entries and shall be a digit in the range of "1" to "9" inclusive

5 2 1 5 2 Size of field position field (LR RP 21)

This subfield shall specify the size in octets of the Field Position field of the directory entries and shall be a digit in the range of "1" to "9" inclusive

5 2 1 5 3 Reserved for future standardization (LR RP 22)

This subfield is reserved for future standardization as an extended entry map Its value shall be the digit "0".

5 2 1 5 4 Size of field tag field (LR RP 23)

This subfield shall specify the size in octets of the Field Tag field of the directory entries and shall be a digit in the range of "1" to "7" inclusive. The value in this subfield shall be the same in all LRs.

5.2.1.6 Alternate forms of counts and field positions

This International Standard provides for both numeric and binary encoding of octet counts and positions

The content of the fifth octet of the DDR Record Length (DDR RP 4) shall signal the use of the alternate forms. The allowed values shall have the following meanings

Value	Signifying that.
"0"-"9"	the character form is used (the flag is the least significant digit of the value)
"B"	the MSOF binary form is used
"b"	the LSOF binary form is used

The forms shall be mutually exclusive and the same form shall be used in the leader/directories throughout the file.

The leaders and directories of the LRs shall have one of the following forms for the contents of the Record Length field, the Base Address of Field Area field, the Field Length field and the Field Position field

a) a character form comprising decimal digits, in which these fields shall contain a right-adjusted decimal integer filled on the left by ZEROs A value of zero in the Record Length field shall specify a record length in excess of 99999

b) a binary form comprising a signed binary integer and, in the DDR Record Length field only, a form flag The first four octets of the Record Length field and Base Address of Field Area field shall be used to record a 32-bit signed binary integer Each instance of an Entry Map shall be limited to a single set of values from the following list 1,1,0,2, 2,2,0,2, 2,2,0,4 or 4,4,0,4 The values of the field length and field position shall be either 8-bit, 16-bit or 32-bit signed binary integers as appropriate depending upon the field sizes specified in the Entry Map The maximum size of DDF record components shall be limited to the maximum value which can be placed in the corresponding control field The unused positions, DR RP 4 and LR RP 16, shall contain a "0"

5.2.2 Logical record directory

The LR directory shall consist of repeated LR directory entries, the field lengths of which shall be specified in the entry map field (see 5 2 1 5) The LR directory shall contain one LR directory entry for each field in the Field Area and shall end with a field terminator

The LR directory entry specifies the location and length of the corresponding field in the field area and shall consist of the subfields shown in figure 6 Each entry shall contain a Field Tag, Field Length, and Field Position in that sequence, and shall consist of $m+n+t$ octets, where

m is the size of the Field Length Field
n is the size of the Field Position Field
t is the size of the Field Tag Field

The LR directory entries shall be in one to one correspondence with the fields and in the same order

The location, length and contents of each directory entry are specified in figure 6 and their meanings are specified in clauses 5 2 2 1 to 5 2 2 3

RP	Name of Field	Length	Contents
$p(i-1)$	Field Tag	t	alphanumeric
$p(i-1)+t$	Field Length	m	digits
$p(i-1)+t+m$	Field Position	n	digits

where $p = t + m + n$ and i = the index of the directory entry

Figure 6 - LR Directory Entry Schematic

5.2.2.1 Field tag field

This field shall contain a Field Tag identifying the corresponding field and shall consist of between one and seven alphanumeric characters as specified in the Entry Map (see 5 2 1 5 4)

5.2.2.2 Field length field

This field specifies the length in octets of the corresponding field (see 5 2 1 6) The length of the field shall include the field terminator

5.2.2.3 Field position field

This field specifies the relative position of the first octet in the corresponding field (see 5 2 1 6) It shall contain a value which when added to the base address of the Field Area shall result in the relative position of the first octet of the field referenced

NOTE 4 - The field position of the first field following the directory is zero

5.3 Logical record field areas

This subclause contains the general requirements of the field areas of logical records Additional requirements for field areas related to the details of data description are found in 6 2, 6 3 and 6 4

The field area of the first logical record (the DDR) shall contain Data Descriptive Fields describing the Data Fields of the file The field areas of the subsequent logical records (the DRs) shall contain the Data Fields as described in the Data Descriptive Fields of the DDR The field tags in the DDR and DR shall be used to associate the Data Descriptive Fields with the corresponding Data Fields

The order of the tagged fields of a field area shall be the same as the order of the tags in the directory of the record

5.3.1 Field area of the DDR

The Field Area of the DDR shall contain one and only one Data Descriptive Field for each data Field Tag that may be used in the Data Records

5.3.2 The field areas of the DRs

The data fields of the Field Area of the DRs shall contain the actual data to be interchanged Each data field shall be associated with an entry in the directory of the DR containing it, and the tag in that directory entry, by reference to the corresponding tag in the DDR and its Data Descriptive Field, may be used to interpret the data field

A data field may be terminated prematurely at the end of a subfield prior to the end of its data description The subfields shall contain the data elements corresponding to the expanded array descriptor contained in the corresponding DDR field Each field shall end with the field terminator In a delimited structure missing data elements shall be represented by successive delimiters. The null values to be associated with the missing data elements of a fixed-length subfield are not defined In a user data field, a single terminal subfield delimiter or a terminal string of adjacent subfield delimiters may be omitted

A data field, described in the DDR, may have zero or more instances in a DR Field Area The directory entries corresponding to missing fields shall be omitted from the directory unless required to describe the data structure of a level 3 DDF (see 6 1 1).

5.3.2.1 User data fields

Each data field shall conform to the Data Descriptive Field associated with the same Field Tag located in the DDR Each data field shall end with a Field Terminator The description of the user data is further defined in 6 3 and 6 4

Character data fields whose termination is determined by a field length, format width, user determined delimiter or field terminator may contain a unit terminator character which shall be treated by an implementation as a user data character.

5 3 2 1 1 Elementary data fields

The data fields of the DR shall contain a single string of octets terminated by the field terminator

5 3 2 1 2 Compound data fields

The data fields of the DR shall contain a single string of octets comprising subfields, terminated by the field terminator.

6 Description of user data types and structures

This clause contains specifications for the description of the user's data fields See clause C.2 and annex D for an informal discussion of the constructs of this clause

6.1 DDR leader fields related to data description

This subclause contains additional specifications for the DDR leader which are related to data description and which are applicable to the entire file

In addition to the fields contained in all LRs, the DDR leader shall also contain meaningful values in the DDR specific fields, as shown in figure 4 and further specified in 6 1.1 to 6 1 5

6.1.1 Interchange level field (DDR RP 5)

This field shall specify the level of the interchange file The content of this field shall be the digit "1", "2" or "3" and these shall have the following meanings

"1"	the file is a level 1 file
"2"	the file is a level 2 file
"3"	the file is a level 3 file

In a level 1 file the Data Fields of the DDR shall contain octet strings which are not differentiated into ISO/IEC 8211 subfields

In level 2 and level 3 files the Data Fields of the DDR shall contain compound data fields which are described by the corresponding Data Descriptive Fields In a level 3 file there is an additional inter-field hierarchical structure

For level 3 files, the tags and fields of the LR shall be in the same order as the preorder traversal sequence of the corresponding fields of the ordered, rooted tree of the LR In the absence of the 0...1 Field Tag, the first Field Tag of the preorder traversal sequence shall be the root node of the tree

NOTE 5 - The clauses which have substantive reference to level specifications are:

Level	Relevant clauses
1	6.3
2	6.4
3	6.2.1.3, 6.2.6, 6.4

6.1.2 Inline code extension indicator (DDR RP 7)

This field shall specify the coded character set methodology used and if additional inline coded character sets are used. SPACE shall mean that there are no character sets other than the basic character set used in the file

NOTE 6 - The other values for this field are specified in clause 7.

6.1.3 Application indicator field (DDR RP 9)

This field contains identification of the user application standard under which a file was produced. SPACE shall mean that there is no identification of the application The value "A" shall specify a reference to another standard (see 6 1.3.1) The values "B" to "Z" inclusive are reserved for future standardization Other values are reserved for private use.

6.1.3.1 Reference to other standards

The value "A" shall mean that another standard is used to specify the content and semantics of the transferred information and the numerical form of the Object Identifier Arcs of the applicable standard will be found in the first subfield of the User Application field (tag = 0. 2, see 6 2.3) followed by an UT when additional user text is required.

NOTE 7 - The syntax and semantics of the Object Identifier Arcs are specified in ISO/IEC 8824-1 Annexes B, C and D.

6.1.4 Field control length field (DDR RP 10-11)

This field shall specify the number of octets of the Data Descriptive Field devoted to data element type and structure codes, delimiters, truncated escape sequence (if any) and other positions reserved for future standardization

The contents of this field shall be:

Level	Non-extended sets	Extended sets
1	"00"	"03"
2 or 3	"06"	"09"

where non-extended and extended sets refer to the use of fieldwise code extensions facilities (see 6.1.5)

6.1.5 Extended character set indicator field (DDR RP 17-19)

This field shall specify the filewise default coded character set used or if fieldwise default character sets are defined SPACEs shall mean that there are no default character sets defined for this file.

NOTE 8 - The values for this field are specified in clause 7.

6.2 Special field tags (tags = 0...0 to 0...9)

This subclause contains requirements for special field tags and the contents of their Data Descriptive Fields which are applicable to the entire file and which are not applicable to an isolated data field.

6.2.1 File control field (tag = 0...0)

The file control field shall be present only in the DDR, shall have the tag 0 ..0 and shall contain the following subfields (see figure 7):

a) the field controls (if any),

b) an optional external file title,

c) for level 3 only, a unit terminator followed by a list of field tag pairs,

d) terminated by a field terminator

Figure 7 - File Control Field Schematic

6.2.1.1 Field control field

These field controls are a special case of the field controls described in 6.3 and 6 4 2 The first four octets for level 2 and 3 files shall contain the value "0000"

NOTE 9 - The printable graphics and escape sequence of the field controls are applicable only to the External File Title as there are no corresponding DR fields

6.2.1.2 External file title field

This field shall specify an optional external file title If present it shall contain a character string which shall be an external descriptive name for the interchange file

6.2.1.3 List of Field tag pairs

In a level 3 DDF, many ordered rooted trees may be described by the use of the list of field tag pairs and the preorder traversal sequence of the tree See annex F for technical background

NOTE 10 - Trees which cannot be represented by this method can be represented by the method of 6 2.6.

The list of field tag pairs is the parent/offspring binary relation of the tags of the DDR which with the preorder traversal sequence of the DDR describe a generic tree structure for the DDF These pairs may be placed in the list in any sequence and shall be logically contiguous Field tags 0...0 and 0...2 to 0...9 inclusive shall not participate in the structure specification

NOTE 11 - The variable hierarchical data structures permitted in the DR are described by supplying the preorder traversal sequences of the data tree and a list of the field tag pairs expressing the parent/offspring logical association between the nodes of the data tree The data structures of the DR shall be derived from the generic data tree by the repetition or deletion of nodes and subtrees The preorder traversal sequence of the generic data tree is supplied by the order of the DDR directory entries The preorder traversal sequence of each instance of a derived data tree in a DR is supplied by the sequence of directory entries in the DR See annex F for further explanation of ordered field tag pairs A forest may be represented as an ordered, rooted tree by creating ordered links from the root node of the first tree to the root nodes of the remaining trees in the proper order.

6.2.2 Record identifier field (tag = 0...1)

The optional Record Identifier Field, if present, is a normal user Data Field, whose content is specified by the user. Each DR shall contain zero or one Record Identifier Fields The content of the Record Identifier Field shall conform to the corresponding DDR Data Descriptive Field

If used, the field corresponding to field tag 0...1 shall be the Record Identifier Field, and except for the LINKS field (tag = 0 . 9, see 6 2 6), shall occur first in the DR directory and shall be the root of any hierarchy (see 6 2.1.3)

6.2.3 User application field (tag 0...2)

If used, the field with the field tag 0...2 shall be present only in the DDR and shall contain application specific information An implementation shall pass this field to the user for processing.

NOTE 12 - The contents of this field are determined by the user and may be used to convey any augmented file description (e.g , file attributes relevant to the interchange) or ancillary file processing controls or application information. See 6.1.3.1 for the use of this field to identify other standards.

6.2.4 Announcer sequence or feature identifier field (tag 0...3)

If used, the field with the tag 0...3 shall be present only in the DDR and shall contain the set of announcer sequences or feature identifiers for extended code set services, see clause 7 2.4 and 7.3 3. Any control characters in this field shall be encoded as their ISO/IEC 6429 mnemonics (see B 2.6) An implementation shall pass this field to the user for processing.

6.2.5 Fields reserved for future standardization

Fields with tags 0...4 through 0...8 shall be reserved for future standardization and shall not be present in the DDR.

6.2.6 Recursive tree LINKS field (tag = 0...9)

NOTE 13 - This field is necessary to describe recursive trees which cannot be described by the list of Field Tag Pairs

In each DR, a recursive tree shall be described by the left- and right-hand links of the corresponding binary tree These links shall be placed in a reserved field having the field tag 0...9, and the following data description (see 6 4)

2*t*00,&LINKS&*LLINK!RLINK&(*intfmt*), where

intfmt	is an integer format control,	= I, I(*d*) or B2*w*,
t	is the data type code,	= 1 for I formats, = 5 for B formats,
d	is the subfield width of an I format,	
w	is the precision of a B format.	

When present, the LINKS field shall be the first field in the DR directory and field area and shall not participate in the tree structure The remaining DR field tags and fields shall occur in the preorder traversal sequence of the tree The left- and right-hand links to the offspring fields shall occur in the preorder traversal sequence and shall comprise the indices of the directory entries of the corresponding fields (see the example of F 2) When there is no link, the corresponding left- and right-link shall be zero When present, the tree structure defined by the LINKS field shall supersede that defined by the field tag pairs

The LINKS field may be used to define non-recursive trees

NOTE 14 - A recursive tree cannot occur in the DDR and the DDR tree is described by its field tag pairs and preorder traversal sequence The pre-order traversal sequence of the corresponding binary tree is the same as that of the ordered, rooted tree

6.2.7 Order of special field tags in the DDR

When present, field tags 0...0 to 0...9 inclusive shall occur first in the DDR directory and in ascending numeric order

6.3 Data descriptive fields in level 1 files

This subclause specifies the content of the Data Descriptive Fields of level 1 files. The user data of the corresponding Data Fields comprises octet strings without division into subfields.

A data descriptive field of a level 1 file shall comprise the following

Field Controls	Data Field Name	FT

A level 1 interchange file may have one Field Control parameter (the truncated escape sequence) whose presence is conditional upon the use of fieldwise default extended coded character sets Its use and values are specified in clause 7.

The Data Descriptive Fields of level 1 files shall contain an optional Data Field Name which serves to identify the data field. The Data Field Name may be any string of characters

6.4 Data descriptive fields in level 2 and 3 files

This subclause specifies the content of the Data Descriptive Fields of level 2 and 3 files. The user data of the corresponding Data Fields comprises subfields.

The data structures specified in this subclause shall be elementary, vector and array structures containing character strings, implicit point, explicit point, explicit point scaled, character mode bit string, bit string, binary forms and mixed data types The data descriptive field shall contain control information, printable graphics, field name, array descriptors and data field format information as specified in 6 4 1 to 6 4 4 and clause 7.

6.4.1 Tabular summary of data descriptive fields of level 2 and 3 files

The data descriptive fields for compound data fields shall comprise the Field Control field, the Data Field Name field, Array descriptors and Format Controls A schematic of a Data Descriptive Field is given in figure 8 Missing elements of the data description shall be represented by adjacent delimiters and a terminal string of adjacent delimiters shall not be replaced by the field delimiter. Table 1 specifies the use of delimiters in Data Descriptive Fields and in Data Fields Table 2 summarizes the data descriptive field components which are specified in this clause and defined further in 6.4 2 to 6.4.4.

Figure 8 - Schematic of Level 2 and 3 Data Descriptive Fields

Table 1 - Delimiters and Their Uses

Delimiter	Print Symbol	Record	Usage
FT [1]	; [2]	DDR,DR	Field terminator
UT [1]	& [2]		Unit terminator
		DR	a) To delimit subfields in a field not specified by a format
		DDR	b) To delimit the Data Field Name
		DDR	c) To delimit the External File Title
		DDR	d) To delimit the array descriptor
"!"	!	DDR	To delimit subfield labels within a vector label
"*"	*	DDR	To delimit the vector labels within a Cartesian label.
"\"	\\	DDR	To delimit the array descriptors of concatenated structures.

[1] These characters are defined by the BCS codes, IS2 and IS1 ((1/14) and (1/15)). With the use of ISO/IEC 10646 codes they are represented by the appropriate multi-octet code (see clause 7 and ISO/IEC 10646 clause 16)

[2] The print symbols, ";" and "&" are the graphic characters used in examples throughout this document to represent the standard delimiters, FT and UT, which are non-printing characters in many systems See 6 4 2.4

Table 2 - Data Descriptive Field Components

Data Structure	RP 0, 1	Data Field Name & Array descriptor & Format Controls;
Elementary	0d	['Name'] & ['subfield label'] & ['fctrl'],
Vector	1d	['Name'] & ['Vector label'] & ['fctrl'],
Array	2d	['Name'] & ['Array descriptor'] & ['fctrl'];
Concatenated	3d	['Name'] & 'Concatenated data description' & ['fctrl'],

The syntax of table 2 is defined below·

a) RP 0 contains the data structure code (see 6.4 2 1), RP 1 = "d" is a data type code (see 6 4.2.2).

b) 'Name' is a data field name (see 6 4 3.1)

c) 'Array descriptor' specifies one of five forms which identifies subfields and structure. These forms are: concatenated structure, numeric array descriptor, Cartesian label, vector label or subfield label (see 6 4.3.2)

d) 'fctrl' specifies a format control (see 6.4.3 3).

e) [ ] specifies the optional or conditional presence of the item.

6.4.2 Field controls

This clause specifies the position, values and meaning of the field control parameters which describe the contents of the DR user data fields The length of the Field Controls field is specified by the contents of DDR RP 10-11. Level 2 and 3 interchange files have the following field control parameters.

6.4.2.1 Data structure code (RP 0)

The data structure code specifies the complexity of the data structure contained in the data field It shall have one of the following values and meanings

0	the dimension of the data structure is zero, i e , a single data item
1	the dimension of the data structure is one, i e , a linear structure
2	the dimension of the data structure is equal to or greater than two, i e , a multi-dimensional structure
3	the data structure is concatenated

6.4.2.2 Data type code (RP 1)

The data type code specifies the type of data in a data field that comprises subfields of a single data type or mixed data types It shall have one of the following values and meanings

Value	The data type is.
0	character string
1	implicit point
2	explicit point
3	explicit point scaled
4	character mode bit string
5	bit string including binary forms
6	mixed data types

6.4.2.3 Auxiliary controls (RP 2-3)

The auxiliary controls further specify the data types contained in the user data field Auxiliary controls are defined for bit field (data type code = "5") and specify binary forms These are specified in clause 6 4 3 3 h) All remaining auxiliary controls are reserved for future standardization and shall be ZEROs signifying that no auxiliary controls are specified

6.4.2.4 Printable graphics (RP 4-5)

The printable graphics are two graphics characters to be supplied by the sender in order that the receiver may use them to replace the standard delimiters for display purposes. They shall represent FT and UT in that order

NOTE 15 - These characters do not affect interchange in any way They should be characters which are seldom, if ever, used in the data and in the absence of information to the contrary should be limited to alphanumeric characters

6.4.2.5 Truncated escape sequence (RP 6-8)

The presence of the truncated escape sequence is conditional upon the use of fieldwise default extended coded character sets Its use and values are specified in clause 7

6.4.3 Data field names, array descriptors and format controls

The use of data field names, array descriptors and format controls are specified below

6.4.3.1 Data field name

A data field name is an optional identifier for a data field and its contents National variant characters or characters from the default set as specified in clause 7 are permitted in data field names

6.4.3.2 Array descriptors

An array descriptor shall consist of a numeric array descriptor or a generic label (i e , a subfield label, a vector label or a Cartesian label) or a compound array descriptor

NOTE 16 - A numeric array descriptor merely defines dimension and extents Generic labels are optional and have the dual purpose of identifying the elements of an array as well as defining its dimension and extents Labelling can only be applied to regular structures, i e , those in which all instances have the same dimension and extents The one exception to this is the "table" an array whose first extent, i e , "row", repeats indefinitely and whose first vector label is, of necessity, null This gives rise to unlabeled "rows" See 6 4 4 for the correspondence of labels and data elements

The following specifications are phrased in terms of "rows" and "columns" but are also specifications applicable to regular structures of higher dimensions The correspondence of the expansion of a Cartesian label to an array is given by the definition of a Cartesian label (see 6 4 3.2 4) and the storage order of an array (see 6 4 4.3)

6 4 3 2 1 Numeric array descriptor

A numeric array descriptor shall be used to specify the dimension and extents of an unlabeled data field

NOTE 17 - The following constructs permit unlabeled arrays whose subfields are identified by position, i e , subscripts.

Fixed array dimensions If an array has a fixed dimension and extents in all data records and a Cartesian label is not required, the Cartesian label may be replaced by a numeric array descriptor which shall be comprised of the dimension of the array followed by the extent of each dimension, all separated by commas.

NOTE 18 - The following construct permits arrays whose instances in data fields have varying dimension and extents

Variable array dimensions In the absence of an array descriptor in the DDR, the DR data field shall be preceded by a positive integer specifying the dimension of an array and a series of positive integers specifying the extent of each dimension The elements of this array description shall be delimited by unit terminators

6 4 3 2 2 Subfield labels

A 'subfield label' specifies an optional single character string which shall identify a single data item A 'subfield label' is a special form of a 'vector label'

6 4.3 2 3 Vector labels

A 'vector label' specifies an ordered set of subfield labels which shall correspond to and identify the items of an ordered set of subfields in the data records and shall take the form, label1!label2!...!labeln A 'vector label' is a special form of 'Cartesian label'

6 4 3 2 4 Cartesian label

A 'Cartesian label' specifies a set of identifiers which shall comprise vector labels forming a Cartesian product the elements of which on expansion shall have an order corresponding to and shall identify the items of an ordered set of subfields in the data records The Cartesian label shall take the form label1!label2!...*labela!labelb!...*...

Cartesian labels, when expanded by the defining conventions (see 6 4 4 2), shall form an array of composite labels corresponding to the elements of the data array The vector labels of a Cartesian label shall provide row and column headings for the appropriate cross section of the array When a field contains a one dimensional structure, the 'generic label' shall be in the form of a corresponding vector label When a field contains a single data item, the generic label is a single 'subfield label'

NOTE 19 - The special cases of a vector containing one subfield and an array containing one vector are allowed.

The first vector label of a Cartesian label may be null permitting the description of a two- (or higher) dimensional array without "row" identifiers A null first vector label shall be indicated by the adjacent delimiters UT and "!" or "\\" and ""!". The use of null individual subfield labels in vector labels is permitted provided all delimiters are present

National variant characters or characters from the default set as specified in clause 7 are permitted in subfield labels The symbols "!" and "*" and "\\" are specific graphic characters used to delimit vector labels and compound Cartesian labels; therefore, "!" or "*" or "\\" shall not appear in a vector label subfield.

A Cartesian label shall not consist of a single vector label composed of a single element containing solely digits and commas

6 4 3 2 5 Description of concatenated structures

NOTE 20 - The intent of this construct is to allow the user to closely associate one or more small arrays with a larger array in order to reduce overhead

A 'compound array descriptor' shall comprise two or more array descriptors concatenated together by "\\" and shall describe the corresponding number of concatenated regular data structures The form of a compound array descriptor shall be.

array_descriptor_1\\array_descriptor_2\\...\\array_descriptor_n

The last and only the last array descriptor of a compound array descriptor label may have a null first vector label The data structure code of a field containing concatenated structures shall be "3"

6.4.3.3 Format controls

The format controls specify the character-by-character or bit-by-bit syntax of the data field The format controls are mandatory for the bit string data type and mixed data types and optional for all other data types when the Field Control Field completely specifies the data format The lack of format controls indicates subfields delimited by the standard delimiter for all data types except for the binary forms which have fixed precisions The format controls are required to specify the sequence and type of the subfields in a mixed data field or to specify the field width in non-delimited subfields or to specify the user delimiters

NOTE 21 - The format controls are hierarchical and can describe irregular structures that cannot be labelled

The format controls shall be pre-delimited by the unit terminator and post-delimited by the field terminator, and shall take the form·

({Y | mY | k(mY,...),...},...)

where

Y implies Z | Z(*)|Z(n) | E,

Z is one of the following

A	signifying character data,
I	signifying implicit-point,
R	signifying explicit-point,
S	signifying explicit-point, scaled,
C	signifying character mode bit string,
B	signifying bit string data,
X	signifying unused character positions

E is one of the following

Btw	signifying the MSOF binary form,
btw	signifying the LSOF binary form

{ }	implies the enclosed expression is to be treated as an entity for purposes of repetition and nesting,
|	implies an alternate choice,
(*), (n)	are subfield width specifications,
*	is an arbitrary user delimiter,
n	is a positive integer specifying the subfield width in octets,
t	is a positive integer specifying the binary form subtype (see 6 4 3 3 h))
w	is a positive integer specifying the precision of the binary form (see 6 4 3 3 h))
m,k	are positive integers signifying the number of times the following data type or group of data types, respectively, is to be repeated,
...	implies repetition of the preceding expression

The use of the format controls is governed by the following rules

a) The order of subfields and their types specified with format controls shall correspond to the data field when the format controls shall be traversed from left to right expanding the nested terms from the left If a repetition factor is not present the value one shall be used If the data field is not exhausted, the format shall repeat from the left-hand parenthesis corresponding to the next to the last right-hand parenthesis not including those parentheses used to delimit subfield width and using the associated repetition factor if any If there is no such right-hand parenthesis, format control shall revert to the first left parenthesis of the format specification

A format, starting and ending with fixed-length bit string subfields, not including binary forms, shall not be rescanned from a single leftmost parenthesis When such a format is to be rescanned the format shall be enclosed in an additional pair of parentheses Two consecutive opening parentheses shall not be used for any other purpose in formats of this type

NOTE 22 - This requirement is necessary to prevent the interpretation of zerofill as data

b) Z - implies delimiting of the DR fields by the unit terminator and, optionally, in the case of the last subfield, by the field terminator.

c) Z(*) - implies the presence of the character, *, as a terminating user delimiter for the corresponding data subfield where * is an arbitrary character other than a digit The delimiter of the last subfield of a data field may be replaced by a field terminator The user delimiter may be a national variant character or a character from the default set as specified in clause 7.

d) Data subfields for I-type, R-type and S-type shall specify a number in a form defined by the ISO 6093 numeric representation forms NR1 (implicit point), NR2 (explicit point, unscaled) and NR3 (explicit point, unscaled) respectively An R-type subfield may contain a fully defined S-type numeric form

e) Data fields containing character mode bit string data (C- type) shall specify bit strings as a sequence of the characters "0" or "1" corresponding to the digits in the bit string represented

f) Fixed-length bit strings (B-type with subfield width specification) must be specified by a format and shall not have subfield delimiters The width of a fixed-length bit subfield shall be specified in bits Vectors and arrays containing fixed-length bit data shall have each subfield adjacent to the preceding subfield The last octet of a fixed-length bit subfield or a series of adjacent fixed-length bit subfields shall be filled on the right with binary zeros The data field shall be terminated by the appropriate field terminator The first of a series of fixed-length bit subfields shall start on an octet boundary The format term, X(0), shall be used only between bit string format terms in order to force the second bit string to start on an octet boundary

g) Variable-length bit strings (B-type without subfield width specification) shall be specified by the following format

Octet 0	Octets 1 to d	Octets d+1 to d+(n+7)/8	Fill
string length count	bit string length	bit string data comprising binary digits	zerofill to complete the last octet
"d"	*n* (digits)	user data	00...0

The string length count shall be a single digit specifying the number of decimal digits in the bit string length The bit string length shall be a sequence of decimal digits specifying the length of the bit string in bits A variable-length bit subfield shall start on an octet boundary

NOTE 23 - Multiple variable-length bit subfields (vectors and arrays) may be specified by the use of an appropriate format statement and variable-length bit subfields may be used in mixed data fields Bit subfields are stored in the MSOF order

h) Binary forms are defined by extensions to bit subfield parameters. The extensions to the field controls and format controls are specified in table 3

Table 3 - Extensions of Bitfield Data Descriptions

Extended Binary Form	Field Controls RP 1-3	Format Term MSOF	LSOF	Precision w = octets
Integer, unsigned	51w	B1w	b1w	1,2,3,4
Integer, signed	52w	B2w	b2w	1,2,3,4
Real, fixed point	53w	B3w	b3w	4,8
Real, floating	54w	B4w	b4w	4,8
Complex, floating	55w	B5w	b5w	4,8

where

precision	is the width of the data item in octets,
field control	is the value to be placed in the associated field control field, RP 1- see 6 4 2 2-3
format term	is the format control for a data item,
w	is an appropriate permitted value of precision.
Integer, unsigned	is a binary integer
Integer, signed	is a two's complement binary integer
Real, fixed point	is a signed integer of precision w/2 followed by a fractional part whi is an unsigned integer of precision w/2
Real, floating point	is defined in IEC 559
Complex, floating point	is defined as a couplet formed of two floating point real numbers precision w in the order real part and imaginary part of the compl number.

A floating point complex number counts as a single entry in any data structure and requires a single format term In the absence of a format descriptor, binary forms shall be stored in the same order as the binary controls if present (see 5 2 1 6 b)), or in the absence of binary controls, in the LSOF order.

NOTE 24 - The integral and fractional parts of fixed real and the real and imaginary parts of complex do not reverse their order when LSOF and MSOF are used.

i) The content of an X-type subfield is not specified and shall be ignored in interchange and is not associated with a label

6.4.4 Order of array descriptors, labels and arrays

This clause specifies the order of numeric array descriptors, Cartesian labels and the storage order of array elements.

6.4.4.1 Order of numeric array descriptors

The general form of a numeric array descriptor shall be d, e(1), e(2),... e(d)

where

d	is the dimension of the array, and
e(i)	is the extent of the i-th dimension.

6.4.4.2 Order of cartesian labels

The general form of a Cartesian label shall be V1*V2*V3*...*Vd,

where

d	is the dimension of the array, A,
V1	is the row vector label having e(1) elements,
V2	is the column vector label having e(2) elements,
Vi	is the vector label of the i-th dimension of the array, A, having e(i) elements,

The order of expansion of the Cartesian label shall be from left to right as follows (((V1*V2)*V3)*... Vd).

Example For the numeric array descriptor 2, 3 ,2 , expansion of the corresponding Cartesian label

row1!row2!row3*col1!col2 would be

row1*(col1!col2), row2*(col1!col2), row3*(col1!col2) = row1!col1, row1!col2, row2!col1, row2!col2, row3!col1, row3col2

6.4.4.3 Storage order of array elements

There shall be a one to one correspondence between the order of the expanded Cartesian labels and the array elements

Example The storage order of the above array having the elements, aij, shall be. a11, a12, a21, a22, a31, a32

NOTE 25 - This implies that the element of the array, aij, is the element from the i-th row and the j-th column Some programming languages express the subscripts in the opposite order.

The storage order of an array of any dimension is given by

$$L(s(1),...s(d)) = 1+\sum_{k=1}^{d}[(s(k)-1)\times\prod_{m=k+1} e(m)]$$

where

A	is an array with the numeric descriptor, e(1), e(2), e(3),... e(d),
d	is the dimension of A,
e(i)	is the i-th extent of A,
L(s(1),...s(d))	is the ordinal of the (s(1),...s(d))-th element,
s(i)	is the i-th subscript of A(1 = first),

and

$$\sum_{k=1}^{m} x(k) \text{ is } x(1)+x(2)+...+x(m)$$

and

$$\prod_{m} x(m) \text{ is } x(m)\times x(m+1)\times\ldots\times x(d)\times x(d+1), \quad x(d+1) = 1$$

7 Use of coded character sets

This International Standard requires the use of the C0 and G0 set of the Basic Character Set throughout a DDF except for the following

a) Subfield labels may be encoded in BCS or, when it has been invoked the UCS-2, two octet encoding form or UCS-4, the four octet form of the ISO/IEC 10646 Basic Multilingual Plane at Level 1

NOTE 26 - Subfield labels have been restricted to level 1 in order to facilitate their use as data identifiers in processing programs

b) External file titles, field names and user delimiters which are discussed in this clause

c) Data fields and subfields of character data type (A-type) which are discussed in 7 1

NOTE 27 - The exclusion of some multiple byte character sets from labels is a departure from ISO/IEC 8211-1985 but is necessary since some multiple byte sets do not include the necessary characters and others may be ambiguous Moreover, processing escape sequences within system identifiers imposes significant overhead for direct access applications When an ISO/IEC 10646 subset has been designated for a file or field, the delimiters, UT and FT, used in a data field are in the ISO/IEC 10646 encoding

This International Standard provides for the use of coded character sets in data fields and subfields of the A type by the methods of ISO 2022 and ISO/IEC 10646 Default coded character set extensions may be announced for a file or separately for each field as described in this clause Subclause 7 1 specifies the announcement of the method of choice and requirements which are common to both methods Subclause 7 2 specifies the use of ISO 2022 and 7 3 specifies the use of ISO/IEC 10646 Both code sets may be specified as filewise and fieldwise defaults or be used as in-line escape sequences The use of appropriate escape sequences will allow the inter-mixture of the code sets in the same file or field

7.1 Announcement of coded character set extension

This field, DDR Leader RP 7, shall specify if coded character sets other than the G0 set of BCS are used in the file The values of this field shall have the following meanings·

SPACE	Only the G0 set of BCS is used and there are no in-line escape sequences
"E"	The BCS is the default character set and in-line ISO 2022 escape sequences may be used.
"h"	Collections 1 and 2 of the ISO/IEC 10646 coded character set form the default character set and there are no in-line Identification of Feature control functions or CSI sequences or ESC sequences are used
"H"	Collections 1 and 2 of the ISO/IEC 10646 coded character set form the default character set and in-line Identification of Feature control functions or CSI sequences or ESC sequences may be used

NOTE 28 - The meaning of the above controls is further modified by the contents of DDR Leader RP 17 - 19 and the DDR Field Controls RP 0-2 (level 1) or RP 6-8 (levels 2 and 3) (see 7 2 and 7 3) which specify filewise and fieldwise default character sets.

7.1.1 Scope of active character sets

The scope of an active character set, i e , one which has been designated and invoked implicitly or explicitly, shall start at the beginning of and terminate at the end of an external file title, a field name or a user delimiter or a subfield of character data type (A-type)

7.1.2 Length of fields and subfields

The length of a field or subfield shall be the octet count including any control characters, escape sequences and multiple octet or octet character encodings

When the use of extended character sets requires the presence of escape sequences, shiftin, shiftout or other control characters, the subfield width shall be indicated by delimiters and fixed width formats shall not be used

7.1.3 Use of multiple octet character sets

This use is subject to the following conditions:

a) The field names shall be written in the designated default set Other escape sequences may occur in the field names and the scope of the invocation shall terminate at the end of the subfield excluding the terminator

b) The delimiters, "!" (EXCLAMATION MARK), "*"(ASTERISK) and "\\"(REVERSE SOLIDIS) of the vector and Cartesian labels shall be written in the BCS or, when invoked, ISO/IEC 10646 BMP at level 1, UCS-2 or UCS-4

c) If multiple octet delimiters are used as the user delimiter, they shall be invoked and their scope shall terminate within the parentheses of the applicable format control

d) The octet counts of a variable-length bit subfield shall be coded as single octets in the BCS

7.2 ISO 2022 coded character set extension

This International Standard permits the use of ISO 2022 for extended coded character sets in data fields, user delimiters, field names, external file titles and labels The use of coded character set extensions as specified in ISO 2022 as default character sets in data fields, field names and for user delimiters shall be limited to sets having three or four octet escape sequences associated with the announcing sequences as specified in ISO 2022 Within data fields , national variant sets and any C0, C1, G0, G1, G2 and G3 character sets may be used in the manner described by ISO 2022 without restriction to length of escape sequences

NOTE 29 - National variant character sets are designated and invoked by use of their registered escape sequences in the same manner as other extended character sets ISO/IEC 10646 sets may be designated from within an ISO 2022 character set

7.2.1 Designation of ISO 2022 coded character sets

An extended character set shall be designated by the presence of its truncated escape sequence in the DDR and shall be invoked by default upon entry into the associated data field at which time the G0 set shall be invoked into columns 02-07 (in the 7-bit environment, 2-7) and the G1 set shall be invoked into columns 10-15 Other invocations shall require the use of an appropriate shift control character If a G0 set has not been designated, the BCS is the G0 set by default The scope of any invocation shall terminate at the end of each data field

7.2.1.1 Use in the 7-bit environment

When an extended G1, G2 or G3 set has been specified as a default character set in a 7-bit environment, the field name, file title, subfield label and user delimiter fields shall begin with the G0 set unless an SO control or other shift control character is present to invoke the G1, G2 or G3 set

7.2.2 Designation of default code set for file

A default character set shall be designated for a file by placing a SPACE or an "E" in DDR Leader RP 7 and the truncated escape sequence in DDR Leader RP 17-19

The truncated escape sequence is the last (n-1) characters of the escape sequence used to specify the extended character set, where n is less than or equal to 4 The sequence shall be left justified and, if necessary, filled on the right with SPACEs If there is no extension specified for a field, these three characters shall be SPACEs

7.2.3 Designation of default code sets for fields

A default character set having an n-character escape sequence for each field shall be designated by

a) placing a SPACE or an "E" in DDR Leader RP 7,
b) placing (2/0)(2/1)(2/0) in DDR Leader RP 17-19,
c) setting the value in the Field Control Length field, DDR Leader RP 10-11, to "03" for a level 1 DDF and "09" for a level 2 or 3 DDF
d) placing the truncated escape sequence in the Field Controls RP 0-2 (level 1) or RP 6-8 (levels 2 and 3) of the appropriate DDR field

7.2.4 ISO 2022 announcer sequence field (tag 0...3)

The ISO 2022 announcer sequences (see ISO 2022 clause 9) for the code extension services used may be supplied by placing the complete list of announcers in the DDR as the contents of the data descriptive field having the tag 0...3 The list of announcers shall be in one of the following formats

a) Announcers for the entire file

The list of announcers shall be the complete three octet sequences concatenated together without further demarcation The field shall be terminated by a field terminator

b) Announcer for each field

The list of announcers shall comprise a list of field tags, each field tag followed immediately by the applicable list of announcers terminated by a unit separator The last announcer shall be terminated by a field terminator

In the absence of this field or in the absence of a field tag from the contents of this field, the announcer sequence is ESC (2/0) (4/4) by default

NOTE 30 - These two cases can be resolved by the presence of the ESC at the start of case a)

7.3 ISO/IEC 10646 coded character sets

This International Standard permits the use of ISO/IEC 10646 for extended coded character sets in data fields, user delimiters, field names, external file titles and labels The default state for ISO/IEC 10646 coded character sets shall be

a) UCS-2, the two octet form,
b) Collections 1 and 2,
c) the C0 and C1 sets from ISO/IEC 6429, and
d) Level 1, no combining characters

Other UCS collections, including levels 2 and 3, may be designated for the entire file or for each field as specified in this clause The collections of ISO/IEC 10646 Annex A shall be specified by their identification numbers using BCS digits which shall be right-justified with left-zerofill

NOTE 31 - The user is referred to ISO/IEC 10646 for a complete description of ISO/IEC 10646 character encoding

The use of the ISO/IEC 10646 multi-octet encoding shall be subject to the specifications of 7 1 3.

7.3.1 Announcement of filewise default character set

The use of a filewise default UCS collection shall be announced by placing the "h" or "H" character in DDR Leader RP 7 and its left-adjusted, zero-filled, ISO/IEC 10646 collection identification in DDR Leader RP 17-19

7.3.2 Announcement of fieldwise default character set

The use of a fieldwise default UCS collection set shall be announced by

a) placing an "h" or "H" character in DDR Leader RP 7,
b) placing three "H" characters in DDR RP Leader 17-19,
c) setting the value in the Field Control Length field, DDR Leader RP 10-11, to "03" for a level 1 DDF and "09" for a level 2 or 3 DDF
d) placing its ISO/IEC 10646 collection identification in the Field Controls RP 0-2 (level 1) or 6-8 (levels 2 and 3) of the appropriate DDR field

NOTE 32 - See ISO/IEC 10646 subclause 17 4 and Annex A for collection identifiers.

The C1 control functions may be used within the data strings (see 7 1)

7.3.3 ISO/IEC 10646 feature identifier field (tag 0...3)

Any additional ISO/IEC 10646 features used in a file (see ISO/IEC 10646 clause 17) may be identified by placing the complete list of the additional feature identifiers in the DDR as the contents of the data descriptive field having the tag 0...3 These additional features shall be the default state of the file or a field and any collections specified shall be in addition to the specification of the data field description The list of identifiers shall be in one of the following formats

a) Feature Identifiers for the entire file

The list of feature identifier sequences shall be the complete ISO/IEC 10646 sequences concatenated together without further demarcation The field shall be terminated by a field terminator.

b) Feature Identifiers for each field

The list of feature identifier sequences shall comprise a list of field tags, each field tag followed immediately by the applicable concatenated list of feature identifier sequences terminated by a unit separator The last identifier shall be terminated by a field terminator.

NOTE 33 - These two cases can be resolved by the presence of the ESC or CSI character at the start of case a)

Annex A
(normative)

ASN.1 and FTAM Registrations

This annex contains definitions of ISO/IEC 8211 document types for storage, and for transfer and access by FTAM Registration of these document type definitions is governed by the procedures of ISO/IEC 9834-2

A.1 Abstract syntax identifier

The abstract syntax identifier for this International Standard shall be {iso standard 8211 abstract-syntax(0)} with the corresponding numerical identifier, {1 0 8211 0}

A.2 Transfer syntax identifier

The transfer syntax identifier for this International Standard shall be {iso standard 8211 transfer-syntax(1)} with the corresponding numerical identifier, {1 0 8211 1}

A.3 FTAM document type definitions

This section contains the FTAM document type definitions for ISO/IEC 8211 files

A.3.1 ISO DDF unstructured document type

A.3.1.1 Entry number DDF-1

A.3.1.2 Information objects

The information objects of the unstructured document type are specified in table A 1

Table A.1 Information Objects in the Unstructured Text Document Type

document type name	{iso standard 8211 document-type(2) unstructured(0)} "ISO DDF unstructured"
abstract syntax names a) asname1	{iso standard 8211 abstract-syntax(0)} "DDF abstract syntax"
transfer syntax names	{iso standard 8211 transfer-syntax(1)} "DDF transfer syntax"
parameter syntax	PARAMETERS ..= null
file model	{iso standard 8571 file-model(3) hierarchical(1)} "FTAM hierarchical file model"
constraint set	{iso standard 8571 constraint-set(4) unstructured(1)} "FTAM unstructured constraint set"
file contents	Datatype1 ..= OCTET STRING

A.3.1.3 Scope and field of application

This document type defines the contents of a file or part of a file, for storage or transfer by File Transfer, Access and Management (FTAM) - ISO 8571, as specified by this International Standard.

A.3.1.4 References

ISO 8571, *Information Processing Systems - Open Systems Interconnection - File Transfer, Access and Management*

A.3.1.5 Definitions

This definition makes use of the terms data element, data unit and file access data unit as defined in ISO 8571-1, and the terms Data Records, unused media space and padding as defined in this International Standard.

A.3.1.6 Abbreviations

ASE	Application Service Element
ASN 1	Abstract Syntax Notation One
DDF	Data Descriptive File
FTAM	File Transfer, Access and Management
P-data	Presentation Data

A.3.1.7 Document semantics

The document consists of one file access unit, which consists only of zero, one or more octet strings The order of these elements is significant The semantics of the octet strings is specified in this International Standard

The document structure takes the form allowed by the FTAM hierarchical file model as constrained by the unstructured constraint set as defined in ISO 8571-2 (see table A1)

There are no size or length limitations imposed by this definition except as prescribed by this International Standard and ISO 8571-2 FTAM-3

A.3.1.8 Abstract syntactic structure

The abstract syntactic structure of the document is a series of ASN 1 data types of the type OCTET STRING as specified in ISO/IEC 8824, whose semantics are defined in this International Standard

A.3.1.9 Definition of transfer

A 3 1 9 1 Datatype definition

The file consists of zero or more values of Datatype1 defined in table A1.

A 3 1 9 2 Presentation data values

The document is transferred as a series of presentation data values Each presentation data value shall consist of one value of the ASN 1 data type "Datatype1", carrying one of the strings of the document

Boundaries between P-data primitives are chosen locally by the sending entity at the time of transmission and carry no semantics of the document type Receivers which support this document type shall accept a document with any permitted transfer options

A.3.1.9 3 Sequence of presentation data values

The sequence of presentation data values is the same as the sequence of the octet strings within the Data Unit of the file

A.3.1.10 Transfer syntax

An implementation supporting these document types shall support the transfer syntax generation rules named in table A1 for all presentation data values transferred. Implementations may optionally support other transfer syntaxes

A.3.1.11 ASE specific specifications

A.3.1 11 1 ISO 8571 - FTAM

This document type is a refinement of FTAM-3 Any implementation capable of supporting FTAM-3 can also transport DDF-1

A 3 1 11 2 ISO/IEC 8211 implementation support

The following clauses apply to any implementation support of DDF-1

A 3 1 11.2 1 The EXTEND operation

The FTAM EXTEND operation shall consist of appending conforming Data Records immediately after the last existent Data Record and padding any unused media space with CIRCUMFLEX characters The semantics of the extend operation are defined in this International Standard

A 3.1.11 2 2 The REPLACE operation

When the replace operation is applied to the root FADU of a "DDF-1" document, the transferred material shall be any "DDF-1" document

A 3.1 11 2 3 Relaxations

Any document requested by a DDF Access System using an "FTAM-3" document type which is a relaxation of the document type under which it was stored shall be supplied as the requested "FTAM-3" document type.

A.3.2 ISO DDF Structured document type

A.3.2.1 Entry number DDF-2

A.3.2.2 Information objects

The information objects of the structured document type are specified in table A 2

Table A.2 Information Objects in the Structured Text Document Type

document type name	{iso standard 8211 document-type(2) structured(1)} "ISO DDF structured"
abstract syntax names a) name of asname1 b) name of asname2	 {iso standard 8211 abstract-syntax(0)} "DDF abstract syntax" {iso standard 8751 abstract-syntax(2) ftam-fadu(2)} "FTAM FADU"
transfer syntax names	{iso standard 8211 transfer-syntax(1)} "DDF transfer syntax"
parameter syntax	PARAMETERS ..= null
file model	{iso standard 8571 file-model(3) hierarchical(1)} "FTAM hierarchical file model"
constraint set	{iso standard 8571 constraint-set(4) ordered hierarchical(5)} "FTAM ordered hierarchical constraint set"
file contents	Datatype1 = OCTET STRING Datatype2 ::= CHOICE{ Node-Descriptor-Data-Element Enter-Subtree-Data-Element Exit-Subtree-Data-Element}

A.3.2.3 Scope and field of application

These document types define the contents of a file or part of a file for storage, for access and transfer by File Transfer Access and Management (FTAM) - ISO 8571, as specified in this International Standard

A.3.2.4 References

ISO 8571, *Information processing systems - Open Systems Interconnection - File Transfer, Access and Management.*

A.3.2.5 Definitions

This definition makes use of the terms data element, data unit and file access data unit as defined in ISO 8571-1, and the terms Data Descriptive Record, Data Record, Field and Field Tag as defined in this International Standard.

A.3.2.6 Abbreviations

ASE	Application Service Element
DDF	Data Descriptive File
DDR	Data Descriptive Record
FADU	File Access Data Unit
FTAM	File Transfer, Access, and Management
HA	Hierarchical All Data Units Access Context
P-data	Presentation Data
UA	Unstructured All Data Units Access Context
US	Unstructured Single Unit Access Context

A.3.2.7 Document semantics

The document consists of file access data units, which consist only of zero, one or more octet strings The order of these elements is significant The semantics of the octet strings is specified in this International Standard

The document structure takes the form allowed by the FTAM hierarchical file model as constrained by the Ordered Hierarchical constraint set, as defined in ISO 8571-2 (see table A 2) and further constrained to a depth of 2 The root node has an associated data unit which contains the DDR as defined in this International Standard There are nodes at level one, each of which identifies a Data Record as defined in this International Standard There are nodes at level two, the data unit of each contains a field and its size as defined in this International Standard

The significance of the Node Names is as follows

Level	Node Name	Node Contents
0	None	
1	Record-index	INTEGER indicating the Record Index
2	Field-index	CHOICE {field-index INTEGER, field-tag-instance SEQUENCE { field tag VisibleString, instance INTEGER}

The data unit contents are as follows

level 0 - the access contexts available are
HA - complete DDF-2 document with structure
US - only the root node data element, i e the DDR
UA - the whole file relaxed to "DDF-1" document type
level 1 - the access context available is HA - the complete record contents
level 2 - the access context available is HA - the field tag, field size and contents

There are no size or length limitations imposed by this definition, except as prescribed in this International Standard

A.3.2.8 Abstract syntactic structure

The abstract syntactic structure of the document is a series of ASN 1 data types of the type OCTET STRING as specified in ISO/IEC 8824, whose semantics are defined in this International Standard.

A.3.2.9 Definition of transfer

A.3.2 9 1 Datatype definition

The file consists of zero or more values of either Datatype1 defined in table A2 or Datatype 2 as defined in table A2, the ASN.1 data type as declared as "Data-Element" in the ASN 1 module ISO 8571-FADU

A.3 2 9 2 Presentation data values

The document is transferred as a series of presentation data values Each presentation data value shall consist of one value of the ASN 1 data type "Datatype1", carrying one of the strings from the document or one value of the ASN 1 data type "Datatype 2"

Boundaries between P-data primitives are chosen locally by the sending entity at the time of transmission, and carry no semantics of the document type Receivers which support this document type shall accept a document with any of the permitted transfer options

A 3 2 9 3 Sequence of presentation data values

The sequence of presentation data values is the same as the sequence of octet strings within the Data Unit in the file.

A.3.2.10 Transfer syntax

An implementation supporting these document types shall support the transfer syntax generation rules named in table A2 for all presentation data values transferred Implementations may optionally support other transfer syntaxes

A.3.2.11 ASE specific specifications

A 3 2 11.1 ISO 8571 - FTAM

This document type may be simplified to the document type DDF-1 The resultant document type contains the same sequence of data values as would result from accessing the structured DDF file in access context UA. That is, only the presentation data values in the abstract syntax name "asname1" are present

A 3.2 11.2 ISO/IEC 8211 implementation support

The following clauses apply to any DDF implementation claiming support of DDF-2

A 3 2 11 2 1 The EXTEND operation

The FTAM EXTEND operation shall consist of appending conforming Data Records immediately after the last existent Data Record and padding any unused media space with CIRCUMFLEX characters The semantics of the extend operation are defined in this International Standard

A 3.2 11 2 2 The REPLACE operation

When the REPLACE operation is applied to the root FADU of a "DDF-2" document, the transferred material shall be any "DDF-2" document.

A.3 2 11 2 3 Relaxations

Any document requested by a DDF Access System using a "DDF-1" document type which is a relaxation of the document type under which it was stored shall be supplied as the requested "DDF-1" document type.

Annex B
(informative)

ISO/IEC 8211 Application Specifications

This International Standard specifies a method for the description of user data fields contained in interchange files and a general method for combining user data fields into user data records Within the specified methods a user must specify the details for a specific application An interchange file set may contain several files and each file may have its own requirements for record content and order as well as the content and order of fields within records The user is responsible for this design and must prepare, validate and maintain the specification This International Standard specifies the occurrence of the data description as it appears in the octet string of the DDR which is intended for machine processing and not for human readability This International Standard anticipates the preparation of DDF files by automated methods and their preparation by an editor is nearly impossible

This International Standard does not specify any requirements for the semantics of user data It does specify methods for the unique identification of each data item and user semantics can be associated with this identifier

This annex describes a manner of supplying the data description specifications for an interchange file set in a manner which is human readable, editable and convertible by computer to a conforming DDR The file also may be subjected to the same validation testing as a DDR or be transformed into a draft specification document

The file described in this annex is not a part of an interchange file set

B.1 Specification of ISO/IEC 8211 Exchange File Sets

An ISO/IEC 8211 exchange set is usually a large data structure which must be specified in great detail The following manner of specification is intended to enable its description in a concise yet detailed manner.

The structural model for an ISO/IEC 8211 exchange set is an ordered rooted tree The general notation for the model is:

```
<Tree root>
 |
 |-<r>-<subtree>
 |-<r>-<subtree>
 |- ...
```

where <tree root> is the root of the tree structure

<r> is the repetition factor of the subtree, <subtree>

In the usual manner of trees, each subtree type may comprise the root of further subtrees The complete specification of an exchange set comprises the following generic subtrees:

```
<exchange set>
 |
 |-<r>-<file type>
 |    |
      |-<r>-<record type>|<file type>
      |    |
           |-<r>-<field type>|<record type>
           |    |
                |-<r>-<field type> <field description>
                |
```

where

<exchange set> is the exchange set name

<r> is a specified repetition factor for a subtree,

= null meaning <r> = 1
= integer, meaning a specific repetition factor
= R, meaning indefinite repetition

<file type>, <record type> and <field type> levels may have an instance of the same type as a subtree (i.e a file can be the subtree of a file, etc)

<file type> = File external file title (see 6 2 1 2)
<record type> = Record <record name>
<field type> = <tag> "(" <structure> ")" <order> " - " <field name>
<tag> = an ISO/IEC 8211 field tag (which associates this field uniquely to its data description)
<structure> = <extents> | <extents> "\\" <structure>
<extents> = <n> | <m*n> | <*n> | <i*j*...>

where

<n>	describes an n-tuple (with n = integer)
<m*n>	describes a 2-D array of m rows and n columns
<*n>	describes a repeating 2-D table with n columns
<i*j*...>	describes a multi-D array with extents i, j,

Note that the intent of <structure> is to provide the user with a concise description of the repetition pattern of the subfields within the field.

<order> = "O" | "O " <tag>, a succinct statement of any intra- or inter-field ordering requirements

where O implies the order of the subfields is significant

O <tag>,... implies the order of the subfields is correlated with the order of the subfields in the field bearing the tag(s), <tag>, ...

<field name> = the ISO/IEC 8211 field name corresponding to <tag>

<field description> = the ISO/IEC 8211 data description for a field The details of field description are given with B.2.

The presentation diagram for ordered rooted trees is:

<tree root>
|
|-<r>-<subtree 1>
|
*-<r>-<subtree 2>
| where * implies<subtree 2> or <subtree 3> but not both
*-<r>-<subtree 3>
|

Note 34 - NOTA BENE In these representations, the preorder traversal sequence rule is: top down, right hand branch first. The parent of any subtree is readily apparent and the parent tags for fields must coincide with the field description The traversal of the tree encounters the repetition factor, perhaps the default of one, of a subtree as the subtree is entered. Further repetition patterns may be indicated in the field descriptions and are summarized in the above <structure> and <order> constructs.

The above tree structures for each file define the order and structure of the data descriptive records (DDRs) and the order and structure of the data records (DRs) Vertical and horizontal lines may be extended and spaces may be introduced into the text for readability To improve the legibility of the specifications for large exchanges, the description may be compartmentalized into subtrees for reasons of clarity. The root of each subtree identifies its nodal position in its parent tree. The documentation provided by this section may be placed as comments in the DFD file described in clause B 2 to yield a complete documentation of an exchange set

NOTE 35 - Due to limitations of page space, except for the most trivial cases the data field descriptions will be disjoint from the tree structure and are related to the tree by the field tag and, optionally, the field name

B.2 ISO/IEC 8211 data field description

This annex presents a method for the representation of ISO/IEC 8211 data description which is logically equivalent to the DDR data description but which is human-readable The data field description (DFD) form is easily prepared and maintained on a simple editor and can automatically be converted to the DDR form for export or verification Use of this form will greatly reduce the effort of ISO/IEC 8211 file design

The method comprises six constructs as follows:

a) File identification
b) Global default specification
c) DDR leader specification
d) One or more instances of DDR data field descriptions
e) An INCLUDE construct
f) A LOCAL construct

This methodology provides an alternative statement of an ISO/IEC 8211 data description, which when converted into an ISO/IEC 8211 DDR may produce a data description which is non-conforming In this event, the specifications of clauses 5 through 7 shall take precedence

This International Standard does not provide specifications for processing a DFD into a DDR

NOTE 36 - A user may find it advantageous to develop the specifications for an exchange file set as a composite file, which after its completion is disaggregated into the component files of the exchange set

B.2.1 General specifications

The DFD is intended as a free form human-readable, computer-parsable format The user is encouraged to use formatting (layout of the definitions) to enhance readability The specifications of this clause contribute to readable and parsable data descriptions

B.2.1.1 End of line

End of line is a system specific sentinel indicating the end of a displayable line Multiple occurrences of end of line shall be logically equivalent to a single occurrence. With the exception of the FOR and REPEAT constructs, and continued quoted strings, each construct shall begin on a new line and end on the same line

B.2.1.2 White space

White space shall be any positive number of SPACE or TAB characters White space shall occur between adjacent terms and values in a construct

Within a Cartesian label construct, white space also includes end of line

B.2.1.3 Comments

Comments shall be introduced by a hyphen and shall be terminated by end of line A comment may occur anywhere except within a quoted string

B.2.1.4 Quoted strings

Quoted strings shall be used to represent tags, label names, field names and other textual quantities Quoted strings shall be delimited by the single quote character. A zero length quoted string shall be represented by two adjacent single quotes An end of line shall not occur in a quoted string. If the character "single quote" is required within a quoted string, it shall be represented by two adjacent single quotes, e g , 'Here"s one'.

The continuation of quoted strings to the next line is accomplished by closing the first segment of the string with a single quote, and placing the appended string on the next line, optionally preceded by white space For instance

```
'This is a string which is'          - this comment does not affect anything
'continued on another line'
```

B.2.1.5 Notation

Generic terms and values are represented by a brief descriptive mnemonic within "<" and ">" Within an extent specification, adjacent numeric values are separated by commas The ellipis, "... " is used to signify repetition of the preceding element. The optional presence of a construct is expressed by enclosing it in square brackets, i e ,"[.]"

B.2.1.6 The INCLUDE construct

The INCLUDE construct has the following form

INCLUDE '<system-specific-file-specification>'

The contents of the specified file shall be logically transferred into the DFD specification file at the location of the INCLUDE construct

NOTE 37 - The effect of the INCLUDE construct is equivalent to textually substituting the construct with the contents of the specified file

An included file may not contain an INCLUDE construct

B.2.1.7 Order of Constructs

The order of constructs shall be:

a) File identification - required
b) DDR leader specifications - tag length required
c) Global default specifications - interspersed as needed
d) Field Specifications - may be repeated
   * Field tag - required
   * Parent tag
   * Printable graphics
   * Escape
   * Array descriptor
   * Format controls
   * THEN
e) LOCAL constructs - unspecified

A construct, except for the INCLUDE construct, shall not be imbedded within any other construct

The order of the special fields 0...0 through 0...9, if present, is specified in 6.2 7 and the order of fields in a level 3 file are specified in 6 2.1 3 and 6 2 6

B.2.2 File identification

This information provides human-readable identification of the interchange system being defined These values are not to be converted into DDR data description fields.

File identification includes the following forms:

<term>	<value>
TITLE	'<file-title>'
AUTHOR	'<file-author>'
DATE	'<file-date>'

NOTE 38 - This file title is not the external file title specified in the DDR

B.2.3 DDR leader specifications

This construct supplies the values for the DDR leader fields which contain global data descriptive information applicable to the entire file The LEADER construct shall reset all unspecified values to their original default values including the global defaults

The canonical form of the DDR leader specification is

```
LEADER
    <term>      <value>
    ...         ...
END LEADER
```

where <term> and <value> are one or more entries from the following table

<term>	<value>	RP	Default	Reference
TAG LENGTH	1-7	23	- -	5 2 1 5 4
LEVEL	1|2|3	5	2	6 1 1
VERSION	0|1	8	1	5 2 1 3
FORM	'SP'|'B'|'b'	4	SPACE	5 2 1 6
INLINE EXTENSION CODE	'SP'|'E'|'H'|'h'	7	SPACE	6 1 2
APPLICATION INDICATOR	'SP'|'<any>'	9	SPACE	6 1 3
ESCAPE	'SPs'|'010'|'<esc>'	17-19	SPACEs	6.1.5

where
"<any>" is a user selected character,
"<esc>" is a truncated ISO/IEC 8211 escape sequence or an ISO/IEC 10646 collection number, and
"SP" for FORM signifies the character form of the leader

Example The escape sequence· ESC 2/1 2/8 2/6 when truncated is encoded as '!(&'

The semantics of the values are specified in the referenced clause The TAG LENGTH which does not specify a default is required The other constructs are required only to change the default.

NOTE 39 - ISO 2022 escape sequences, often interpreted as unsigned binary integers, are members of a graphics character set easily represented by most editors and displayed on many devices ISO/IEC 10646 code set collections are designated by integers in the range of 0 to 999 The octets of ISO 2022 truncated escape sequences and collection numbers may be represented by a quoted string or decimal number respectively

B.2.4 Global default specification

The following constructs provide default values for the Field Controls field They shall be applied from their point of definition until their redefinition or the end of file

<term>	<value>	default	clause
DEFAULT PARENT	'<tag>'	'0 .1'	6.2.1.3, B.2 5
DEFAULT PRINTABLE GRAPHICS	'<ab>'	',&'	6.4.2.4
DEFAULT ESCAPE	'<esc>'	SPs	6.4.2.5, B.2.5

where
<tag> is a Field Tag value
<ab> are the Printable Graphics characters
<esc> is a truncated escape sequence or collection number, see B 2 3

B.2.5 Data field specifications

These constructs supply the information (see 6 3 and 6 4) necessary to form a DDR data description field The constructs are presented as a canonical form followed by several special cases of the canonical form, which are applicable to simple field structures and formats Field descriptions shall be supplied in the order they are to occur in the DDR

The canonical form of the data field description is given below.

```
FIELD '<tag>' '<field_name>'                    (see 5 2 2.1, 6 3, 6 4 3 1)
```

The DDR field name is optional and may be omitted

NOTE 40 - For the '0...0' tag, the field name shall be used to specify a default external file title in the DDR file The only additional field information that may be required for this field is ESCAPE

PARENT '<tag>' Default = DEFAULT PARENT

NOTE 41 - The parent tag information is used to construct the DDR list of tag pairs.

PRINTABLE GRAPHICS '<ab>' Default = DEFAULT PRINTABLE GRAPHICS

ESCAPE '<esc>' see B.2 3 Default = DEFAULT ESCAPE

NOTE 42 - The presence of the preceding three constructs is optional When one is omitted, the current global default shall take effect.

An array descriptor construct comprising one or more of the following

1) A structure code

STRUCTURE CODE <d> (see 6 4 2)

where <d> is the field structure code and the default is "0"

NOTE 43 - This construct is used only when the structure code can not be determined from a numeric descriptor or a Cartesian label

2) A numeric descriptor

NUMERIC DESCRIPTOR <ex1>,<ex2>, ... (see 6 4 3.2 1)

where <ex1>,<ex2>,... are the extents of the array

NOTE 44 - This construct is used only when the structure code can not be determined from a Cartesian label The extent information does not include the dimension of the array

3) A Cartesian label

FOR 'row1' 'row2' ... BY 'col1' 'col2' ... BY ... BY 'page1' 'page2' ... (see 6.4.3 2 4)

NOTE 45 - The 'BY's within the Cartesian label construct are equivalent to the "*"s within the DDR Cartesian label The whitespaces between the subfield labels are equivalent to the "!"s of the DDR vector labels

The REPEAT Construct.

The following construct specifies the DDR hierarchical format controls

REPEAT <repeat-group> (see 6 4 3 3)
<m> <fmterm>
. ...
REPEAT <repeat-group>
<m> <fmterm>
. ...

END REPEAT
<m> <fmterm>
. ...
END REPEAT

where
<repeat-group> is
a) a repetition factor, or
b) "TO END" , or
c) "BIT FIELD"
<m> is a repetition factor and <fmterm> is an elementary format term as described in 6 4 3.3

There shall be only one <m> <fterm> pair per display line and when <m> is omitted the default value shall be one.

REPEAT BIT FIELD (see 6.4.3.3 a))

This construct provides for the repetition of formats used with bit fields and shall introduce the supplemental paired parentheses into the format control string. The matching END REPEAT must be immediately before the END FIELD

REPEAT TO END (see 6.4 3.3 a))

This construct provides for the repetition of a set of terminal format terms and inserts the appropriate paired parentheses at its location in the format description and at the end of the format The matching END REPEAT must be immediately before the END FIELD

NOTE 46 - The format generated is used to set the data type code (6 4 2 2) and under some circumstances an implementation may compress a format and eliminate unnecessary parentheses and terms. When format controls are optional (6 4 3.3) they may be omitted

THEN (see 6 4 3 2 5)

NOTE 47 - The 'THEN' construct is equivalent to the "\" characters in the DDR concatenated structure description, and if present shall be followed by appropriate additional field constructs, i e , an array descriptor and a format control

END FIELD

NOTE 48 - The 'END FIELD' construct terminates the field description

B.2.6 Special forms of field constructs

The following special forms shall be recognized for fields in which there is a correspondence between the array elements of the last vector label of the Cartesian label and the repeating format. They may be used in combination when appropriate

B.2.6.1 Null first vector label

A repeating row table with a null first row vector label (see 6.4.3.2 4) takes the form

```
FOR BY 'col1' 'col2' ... BY ... BY 'page1' 'page2' ...
```

B.2.6.2 Correspondence of format and last vector label

The following form may be used for constructs of 2-D, 1-D and 0-D structures. It may also be used for 2-D structures with a null first vector label

```
FOR 'row1' 'row2' ... BY

  'col1'  <fmterm1>
  'col2'  <fmterm2>
   ...      ...
```

The following encoding rules shall apply:

1) The FOR shall always be present and shall not occur on the line with the first label-format term
2) There shall be one pair of label-format term values per display line
3) A null subfield label is encoded as a null string
4) A null row vector label shall be encoded by omitting the row labels and including the "BY".
5) An n-tuple shall be encoded by omitting the row labels and the "BY"

B.2.6.3 Special DDR tagged fields

The contents of the special DDR tagged fields, 0...2 and 0...3, (see 6.2 3-4) shall be provided by one of the special forms:

```
FIELD '0. 2'
    CONTENT     '<special_field_contents>'          see 6 2 3
END FIELD

FIELD '0 . 3'
    ESCAPE      '<control sequence>'                see 6.2.4
END FIELD
```

where 'control sequence' may be an ISO 2022 announcer sequence or an ISO/IEC 10646 identification of features as specified by the appropriate standard and encoded as quoted strings using the mnemonics of ISO/IEC 6429 to represent control characters, e.g , 01/11 as 'ESC' and 09/11 as 'CSI'.

B.2.7 Special constructs

This construct provides for the use of system dependent constructs in a manner which will not compromise the standard data description constructs of this Annex Any local construct shall begin with the text "LOCAL " and shall not be used to replace any of the specified functions of this Annex A system may ignore any local construct

NOTE 49 - Examples of this construct to pass local processing information associated with a file to the export software might be:

```
LOCAL DROPPED LEADER   d
LOCAL INPUT MODE   'a'
```

B.3 Examples of exchange set specification

The following is an example of the data description methodology of this annex and explanatory comments are included in the text

```
TITLE          'Exemplar ISO/IEC 8211 Revised Data Field Description'
AUTHOR         'A A Brooks'
DATE           '1992-06-20'

- The following are the specifications for the DR records:

- File  DDFDFD DEMO FILE
- |-R-- Record  ISO/IEC 8211 data exemplar

- Record. ISO/IEC 8211 data exemplar

- 01 (1) - RECID
- |-R-- TX (1) - TEXT
- |-R-- E0 (1) - CHARACTER
- |
- |-R-- V0 (n) - VECTOR/CHARACTER
- |-R-- Vx (n) - Vector/CHARACTER
- |    |-R-- V1 (6) VECTOR/IMPLICIT POINT
- |    |-R-- V6 (6) VECTOR/MIXED DATA TYPES
- |
- |-R-- A0 (5*2) - CHARACTER ARRAY
-      |-R-- A1 (3*2) ARRAY/IMPLICIT POINT
-      |-R-- A6 (*2) ARRAY/MIXED DATA TYPES
-      |-R-- A7 (3*2) ARRAY/NUMERIC ARRAY DESCRIPTOR
-           |-R-- A8 () ARRAY/NUMERIC ARRAY DESC IN DR
-           |-R-- A9 (2\\2*3) CONCATENATED VECTOR\\ARRAY

-----------------------------------------------------
-The data description of this example is similar in part to that described in the example
-of annex E 5  Several default values are explicitly declared for illustration and so are
-indicated  by a comment  A few parameters, not used by this example are illustrated
-by annotated comments
-----------------------------------------------------

LEADER
  TAG LENGTH              2          - required
  LEVEL                   3
  VERSION                 1          - same as default
  FORM                    'b'
  INLINE EXTENSION CODE   ' '        - same as default
  APPLICATION INDICATOR   ' '        - same as default
  ESCAPE                  ' '        - same as default
END LEADER
```

```
DEFAULT PARENT                  '01'            - same as default
DEFAULT PRINTABLE GRAPHICS      ';&'            - same as default
- DEFAULT ESCAPE                '  '            - same as default but not applicable to this file

- The following field is the external file title field and provides a default external file title
FIELD '00'   'DDFDFD DEMO FILE - LEVEL 3'
END FIELD

- The following field is the ISO/IEC 8211 record identifier field
FIELD '01'   'RECID'                             - field tag and field name
FOR
  ''   A                                         - ISO/IEC 8211 record identification
END FIELD

- The following field is the DDR user application field
FIELD '02'                                       - this DDR field has no field name
 CONTENT      'ISO/IEC 8211 - example'           - defines field contents
              '(example of string continuation)'
END FIELD

- The remaining fields are user data fields
FIELD 'TX'   'TEXT'
PARENT '01'
FOR
  '' A                                           - elementary field without a label
END FIELD

FIELD 'E0'   'CHARACTER'
PARENT '01'
FOR
  'redundant' A                                  - elementary field with redundant label
END FIELD

FIELD 'V0'   'VECTOR/CHARACTER'
PARENT '01'
STRUCTURE CODE 1                                 - structure code in lieu of labels
 REPEAT
   A( )                                          - user delimiter, SPACE
 END REPEAT
END FIELD

FIELD 'Vx'   'Vector/CHARACTER'
PARENT '01'
STRUCTURE CODE 1
 REPEAT 2                                        - nested format groups
   REPEAT 3
    2 A( )                                       - user delimiter, SPACE
      A(%)                                       - user delimiter, %
   END REPEAT
   A(7)
 END REPEAT
END FIELD

FIELD 'V1'   'VECTOR/IMPLICIT POINT'
PARENT 'Vx'
FOR
  'int 1'    I(4)                                - Cartesian label element & format term
  'int 2'    I(4)                                - fixed length integer
  'int 3'    I                                   - variable, default delimiter, (1/14)
  'int 4'    I
  'int 5'    I(#)                                - variable, user delimiter, "#"
  'int 6'    I(#)
END FIELD
```

```
FIELD 'V6'  'VECTOR/MIXED DATA TYPES'
PARENT 'Vx'
PRINTABLE GRAPHICS '#$'                    - non-default printing graphics
FOR
  'text 1'   A(4)
  'int 1'    I(4)
  'text 2'   A                             - default delimiter, (1/14)
  'int 2'    I
  'fp#'      S(%)                          - user delimiter, "%"
  'cmbf'     C
END FIELD

FIELD 'A0'  'CHARACTER ARRAY'
PARENT '01'
NUMERIC DESCRIPTOR 5,2                     - fixed extents
 REPEAT
  A( )                                     - SPACE used as delimiter
 END REPEAT
END FIELD

FIELD 'A1'  'ARRAY/IMPLICIT POINT'         - with row labels
PARENT 'A0'
FOR 'row 1' 'row 2' 'row 3' BY
  'col 1'    I(4)
  'col 2'    I(4)
END FIELD

FIELD 'A6'  'ARRAY/MIXED DATA TYPES'       - no row labels
PARENT A0
FOR BY
  'col 1'   A(4)
  'col 2'   I(4)
END FIELD

FIELD 'A7'  'ARRAY/NUMERIC ARRAY DESC'     - no labels
PARENT 'A0'
NUMERIC DESCRIPTOR 3,2                     - fixed extents
 REPEAT
   A(4)
   I(4)
 END REPEAT
END FIELD

FIELD 'A8'  'ARRAY/NUMERIC ARRAY DESC IN DR'      -no labels
PARENT 'A7'
STRUCTURE CODE 2                           - numeric descriptor will occur in data field
 REPEAT
  I(4)
 END REPEAT
END FIELD

- The following field has a vector catenated to an array
FIELD 'A9'   'CONCATENATED VECTOR\ARRAY'
PARENT 'A7'
FOR   'element 1'  'element 2'
 REPEAT
  A($)
  I(5)
 END REPEAT
THEN
 NUMERIC DESCRIPTOR 2,3                    - a fixed size 2x3 array
 REPEAT TO END                             - forces format repetition from here to end
   I(6)
 END REPEAT
END FIELD
```

```
- The following are host system parameters used to control the
- export software. Only the term "LOCAL" is a standard construct.
LOCAL INPUT MODE  'C'
- LOCAL DROPPED LEADER 1                    - not applicable to this example
```

Annex C
(informative)

Informal Introduction to ISO/IEC 8211

ISO/IEC 8211 is an adaptation of ISO 2709 using its basic record structure and replacing its bibliographic data description by a general purpose data description, removing the magnetic tape limitations and, in this version, introducing binary octet counts

In their simplest terms ISO/IEC 8211 constructs are divided into two groups

1) Logical record and field constructs - Those necessary to reading an ISO/IEC 8211 logical record and to the isolation of the user data fields in the receiving system See clause 5

2) Data description and identification constructs - Those necessary to the isolation of each user subfield with its associated description and identification See clause 6 and Annex D

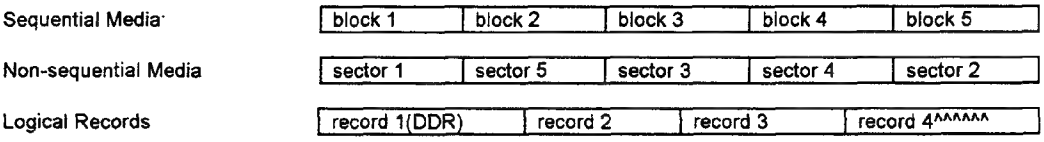
The result is a two layer standard in which the reading of records and fields as strings is quite independent of their interpretation as data description or data The current document is organized in the same manner This division of constructs lends itself not only to a quicker understanding but also to a better organization of implementation software. This annex provides the user with an informal explanation of the methodology

The file or record import step is quite simple, the processing can be extremely reliable and is essential to all implementations The disassembly of fields into subfields is related to the complexity of the user's data and its data description Users need to master only the complexity required to describe their data and not the (usually) much greater complexity of the most general ISO/IEC 8211 application

C.1 ISO/IEC 8211 File, logical record and field constructs

C.1.1 Media record constructs

ISO/IEC 8211 File Schematics

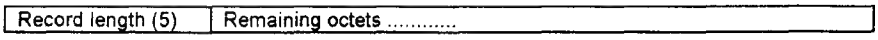


ISO/IEC 8211 files may be transported by any means which preserves the integrity of the octet string of the file or any substring of the file requested by the receiver When ISO/IEC 8211 logical records are written, blocked and spanned, into fixed-length media records (blocks, sectors or packets), the unused portion of the last media record is to be filled with CIRCUMFLEX characters, (5/14) This results in a file of fixed-length media records with the file termination under the control of ISO/IEC 8211 and independent of the media, except for the trivial case where a system end-of-file signal is received while trying to read a logical record This practice minimizes the probability of any problems with current and future media including packet transmission and places minimal demands on the host receiving system

A host receiving system is only required to read a fixed length media record as the ISO/IEC 8211 software can, when necessary, be responsible for the unblocking of logical records This does not preclude a receiving system from reading the variable-length logical records if it is able to do so

C.1.2 Logical record constructs

Logical Record Construct Schematic



The ISO/IEC 8211 logical record is a trivial construct a five octet record length followed by an octet string The count includes its own length The last octet of the record is a field terminator The record length, when added to the starting position of the record, also points to the start of the next record and can be used to rapidly skip records or to produce a simple random

access index to the logical records. ISO/IEC 8211 supports both character and binary representations of record length and other octet counts. Long records in excess of 99 999 octets are supported

C.1.3 Logical record structure

Logical Record Structure Schematic:

Leader(24)	Directory ...	Field Area ...

The ISO/IEC 8211 logical records comprise three constructs which serve to locate the fields within a logical record These constructs are:

1. The leader - Twenty four octets comprising fixed length subfields whose contents are necessary to reading the directory entries and locating fields

2. The directory - comprising a variable number of fixed-length subfields, called directory entries, whose contents are necessary to identify and to locate a field.

3. The field area - comprising a variable number of variable-length fields whose identity, location and length are described by the directory entries The field contents need not be known to the importing system at this time All variable length fields are followed by a field terminator

At the conceptual level, all ISO/IEC 8211 logical records have these three constructs although there are two simple variations which will be discussed later

C.1.3.1 Leader (RP 0 - 23)

Leader Schematic:

Record length(5)	Leader identifier(1)	Version identifier(1)	Base address of data(5)	Directory map(4)

Directory Map Sizes of:

Field length	Field position	reserved	Field tag

NOTE 50 - RP is the position relative to the start of a field or record.

In addition to the record length described above (RP 0 - 4), a leader comprises four subfields

1. Leader Identifier - RP 6 - Indicates the variant of the logical records

2 A version identifier - RP 8 - SPACE = ISO/IEC 8211 1985, "1" = this version

3. Base Address of Data - RP 12 -16 - Starting position of the field area = Number of octets in the leader and directory for the specific record.

4 Directory Map - describes lengths of directory entry subfields,

RP 20 - Length of Field Length Subfield,
RP 21 - Length of Field Position Subfield,
RP 23 - Length of Field Tag Subfield

Additional DDR leader subfields related to data description will be described later

C.1.3.2 Directory

Directory Schematic:

Entry 1	Entry 2	Entry 3	Entry 4	Entry 5

Directory Entry Schematic:

Field tag	Field length	Field position

The directory has one directory entry for each field in the field area and is, by the field position, associated with a unique field. Each entry comprises the following subfields:

1 Field tag - identifies the associated field
2. Field length - the octet count for the field
3 Field position - relative to the Base Address of Data

The directory, being a variable length field, is terminated by a field terminator.

C.1.3.3 Field area

Field Area Schematic:

Field 1 #	Field 2 #	Field 3 #	Field 4 #	Field 5 #

The single field area starts immediately after the directory and contains one field for each entry in the directory The fields are contiguous and in the same order as the directory entries. Each field is terminated by a field terminator ("#").

The fields in the DDR, i e , the first record, contain the ISO/IEC 8211 data descriptions and, in the DRs, they contain data Each directory entry, by its tag, provides a unique association of a Data Field in a DR with a Data Descriptive Field in the DDR and thus allows a receiving system to decode the data field The field length and the field terminator allow an integrity check.

C.1.4 File characteristics and processing

The above concludes the essentials of the logical record constructs of an ISO/IEC 8211 file. An understanding of these constructs provides a basis for understanding of the two variants as well as the role of user data and data description in ISO/IEC 8211 files The most important point is that the above constructs, related to the basic import step, are NOT dependent on the content of fields, i e., the nature of the data or the data description The file has the following characteristics:

1 The constructs are simple and totally defined by ISO/IEC 8211.

2 The values of parameters are dependent only on octet counts of the data fields and not on the nature of the data. On export, these values should be computed by the exporting software from the octet counts and the control parameters of the exported files should be error free even if the field contents are in error

3 There is sufficient redundancy to detect errors, i e., inconsistent octet counts and field terminators, redundant directory information, et cetera

4 Software, at this level, is simple and should be totally reliable. Short of hardware errors or exceeding software array size, import should be certain.

ISO/IEC 8211 Record/Field Import Algorithm: The following algorithm carries out the fundamental import step; i.e., moving the fields from the media into the recipient's computer system. Any further processing is performed within the recipient's system and, strictly speaking, is not a part of the import step

1. get five octets (on end of file, stop), determine the record length,
2 get the remainder of the logical record,
3 determine the base address of data and entry map; determine the values in the size of tag, field length, and field position subfields,
4 process next directory entry (field tag, length and position), if none, goto 1.
5 process the corresponding field, goto 4.

The function "get" may be a variable octet READ or a subroutine which unblocks octets from the fixed-length media records.

The above algorithm can be implemented in most high level languages by a very simple program. The following example implements the algorithm for the character mode octet controls using Fortran.

```
      CHARACTER A*32767, FMT*9, TAG*7
      OPEN(1, FILE='DDF', STATUS='OLD', FORM='BINARY')
      OPEN(2, FILE='IMPORT')

  100 READ(1, END=300) A(1:5)
      READ(A(1:5), '(I5)', ERR=300) NA
      READ(1) A(6:NA)
      READ(A(13:24), '(I5,3X,2I1,1X,I1)') IB,NL,NP,NT
```

```
    FMT='(A,I'//CHAR(NL+48)//',I'//CHAR(NP+48)//')'
    ID=25

200 READ(A(ID:ID+NT+NL+NP-1), FMT) TAG(1:NT), LF, IF
    WRITE(2, '(A,1X,I5,1X,A)') TAG(1:NT), LF, A(IB+IF+1:IB+IF+LF)
    ID=ID+NT+NL+NP
    IF (ID LT.IB) GOTO 200
    GOTO 100

300 END
```

Where the undeclared INTEGER variables are

NA	the record length
IB	the base address of data
NL	the size of the field length field
NP	the size of the position field
NT	the size of the field tag field
ID	the index to a directory entry

The OPEN command for the DDF file may require variation for some hosts

C.1.5 Variant logical records

The variants of ISO/IEC 8211 logical records serve two purposes

1 Enabling very long records
2. Reducing unneeded overhead

C.1.5.1 Long ISO/IEC 8211 records

In the character mode, the maximum record length that may be recorded in the record length subfield is 99,999 octets (The binary mode limit is much higher.) For longer records, this subfield is set to 0 and the import software must employ a slight variation in the import logic, i e , rather than getting the entire record at one time, get the leader and directory and then get each field from the medium as the algorithm steps through the directory entries This logic can also be used to reduce the array size required to read smaller records in a very small computer.

C.1.5.2 Fixed-formats - repeating leaders and directories

ISO/IEC 8211 files derived from very regular source data may have data records in which the DR leader/directories are identical. In order to conserve space, ISO/IEC 8211 permits the following:

• On export, place a "D" in RP 6 of each data record until reaching the first data record having the repetitious leader/directory. Place an "R" in RP 6 of this record For the remaining data records, write only the data area into the file, terminating the file in the normal manner

• On import when sensing the "R", retain the leader/directory and for the subsequent records, get only the data area and concatenate it to the retained leader/directory

C.1.6 ISO/IEC 8211 End-of-data conditions

At end-of-data, software processing ISO/IEC 8211 files experiences a system generated end-of-file or encounters a circumflex in the first illegal location These locations are at:

1) LDR RP 4 in files with all leaders present (LDR RP 6 = "D"),
2) The end of the first unused data field in files with dropped leader/directories (LDR RP 6 = "R")

Testing for these conditions is simple and can be confirmed, if desired, by testing for circumflex from the end of the media record back to the end of the last used ISO/IEC 8211 field Detection of end-of-data is media independent, reasonably robust in cases of improper padding and does not require the double processing necessary to place a record count in the first record or establish some other anticipatory signal

C.1.7 Summary of the logical record and field constructs

Even with the two variants, ISO/IEC 8211 files retain the desirable simple characteristics described in C 1 4. Export and import should have a high degree of reliability Obviously, the user can mismatch a file and software at the host system level but these events are beyond the scope of a standard

C.2 Data description and identification

The second component of the ISO/IEC 8211 transport model comprises data description and data identification In order to focus attention on the model, only the general constructs of data description will be discussed in this annex with details presented in annexes D, E and F

C.2.1 Components of data description

In its most general form, in addition to data identification, ISO/IEC 8211 data description has two components·

1) the description of subfields which have data type and data extent, and
2) the description of fields which may have internal structure and inter-field structure

A data field consisting of subfields having the same data description can be said to have a data type, i e , that of its subfields A portion of each data descriptive field, i.e , the field controls, is devoted to a synopsis of the data description and for some simple descriptions is all that is necessary

ISO/IEC 8211 data description comprises the following

C.2.1.1 Data extent

At the subfield level, data subfield extent can be either fixed or variable For a fixed extent a subfield width is given in the DDR format controls and for a variable extent the end of a subfield is determined by the presence of a subfield delimiter in the data field, which is specified in the format controls for the field

C.2.1.2 Data position

At the subfield level, the position of each subfield is determined from the subfield extents by simultaneously traversing the data field and the data description If all the subfields have the same fixed width the position of the i-th subfield can be computed without traversing the preceding subfields.

C.2.1.3 Data structure

Data structure is a characteristic of a field indicated in the DDR field controls with further details supplied by an array descriptor The label form of the array descriptor is closely related to the identification of subfields Elementary data items, n-tuples, multi-dimensional arrays and sets are accommodated The arrays may be regular or irregular Instances of unlabeled arrays may have variable dimension and extents throughout a file. Unlabelled linear structures, e g., sets, lists, can have an indeterminate number of items

Special arrays such as diagonal matrices, triangular matrices, et cetera need to be transferred as an application-specific structure

C.2.1.4 Data type and syntax

Data type and its implied syntax is specified in the format controls in the DDR Both character and binary representations are accommodated: five character mode data types, bit strings and five binary data types each with several precisions.

Extended character sets may be defined on a filewise or fieldwise basis with provision for inline sets ISO 2022 and ISO/IEC 10646 methods are accommodated

C.2.1.5 Intra-field tree structure

ISO/IEC 8211 supports an ordered, rooted tree structure in which the nodes are the data fields The tree structure is described by the preorder traversal sequence and the parent-offspring binary relation or by left- and right- hand links For most trees the overhead is negligible The data trees can have repeating and missing subtrees. See annex F

Except for inter-field tree structure, the data description of one field is independent of other fields and the data descriptions reside in the DDR quite independently of the data fields in the DRs This makes it feasible to assimilate the data descriptions prior to data field processing converting it to more effective forms if desired and retaining it for future use if desired.

C.2.2 Data identification

Data identification at the field level consists of the ubiquitous field tag and an optional field name.

At the subfield level, data items may be identified by labels (which also contribute to structure definition) Only regular structures, those whose instances have constant dimension and extents, can be labelled This is a logical constraint, not an ISO/IEC 8211 constraint, it is not possible to à priori enumerate labels for an indeterminate number of items The labeling of multi-dimensional structures employs a Cartesian cross product, limiting the identification to rectangular structures.

There are quasi-regular structures which can be partially, but meaningfully, identified The 2-d example of this form is a rectangular array with indeterminate repetition of the unlabelled rows (a table) Higher dimensioned structures have analogous forms

ISO/IEC 8211 also supports the labelling of concatenated regular structures.

C.2.2.1 Application semantics

The data description capabilities of ISO/IEC 8211 relieve the application of considerable documentation of data syntax The identification of fields and subfields enables the application to uniquely associate the application semantics with a data item The ISO/IEC 8211 data description can be expressed in a convenient one tag/subfield per line format which can include subfield semantics and other comments This format can be converted automatically into the data descriptive field form See annex B

C.3 File and record contents

Other than placing the data description in the DDR and the data in the DRs, ISO/IEC 8211 does not attempt to define what fields make a user's record nor what records make a file These are strongly application dependent and left to the file designers who can define the content of each file by itemizing the records in order with proper attention to repetition factors and to define the content of each record by itemizing the field tags in a similar manner A simple tree diagram usually suffices for even complex exchanges An application interchange standard can add specifications as needed

C.4 Binary directories

Binary mode directories introduce no new concepts, only a different representation, and were intended to facilitate the importation of selected data from large files on large direct access devices. They can also reduce the storage overhead Their increased efficiency may be more anticipated than real for three reasons:

1 Except for very slow processors and very big directories, the directory processing time for the character mode is small compared to the read time

2 Several computers do not store integers in the standard binary forms used

3. Direct access indices can be constructed to eliminate much of the need for repetitive directory processing

Binary mode directories should probably be used only where a distribution project can be assured of the nature of the data use.

Annex D
(informative)

Introduction to ISO/IEC 8211 Data Description

This annex contains an informal introduction to ISO/IEC 8211 data description

D.1 Data description - user data

ISO/IEC 8211 data description is mainly concerned with the interior of the user's data field and its complexity varies considerably with the interchange file level and the application The following are guidelines for the beginner

You must fulfill Sublauses 6 1 1 and 6 1 4

If you do not have	You can/may skip.
1) Extended coded character sets (1)	Subclauses 6 1 2, 6 1 5, 6 2 4
2) Application specifics (1)	Subclause 6 1 3, 6 2 3
3) Ordered, rooted tree structure (1)	Subclause 6 2 1 3
External file title	Subclause 6 2.1
4) Complex fields or bit fields	Subclause 6 4 3 3
Arrays	Subclause 6 4 3 2, 6 4 4
Labels	Subclause 6.4 3 2
5) Subfields	Subclauses 6 4

Some of the above are frequently not needed

If you do not have any of these, then you have a level 1 file, the data are strings requiring no description beyond that given in the directory entry (tag, length, position) and perhaps an optional field name Mastering data description in the reverse of the above order simplifies the task.

D.2 Consistency of data description and data - validation

Although the processing of data description and identification is not as foolproof as that of the logical record structure, an exporter in possession of complete ISO/IEC 8211 software can validate an interchange file verifying that the data description is syntactically correct and semantically consistent. The syntactic consistency of the data description and the data can also be automatically validated The data can be in error, outside of reasonable ranges or semantically wrong but these questions are in the realm of the application, not a general purpose interchange standard.

D.2.1 Complexity of data description

Simple user data requires only simple data description Conversely, complex user data requires complex data description and an interchange standard would be deficient if it could not meet user needs The complete ISO data description facility is complex albeit well defined The user need master only that portion suited to their data Data description is a tool and like most tools requires some dexterity on the part of the user There appears to be no easy remedy for the file designer This section introduces data description by level of complexity and not with exhaustive detail

ISO/IEC 8211 permits the use of character sets other than BCS The use of these sets appears complex to the non-user They are also of little interest to the non-user who need only place SPACEs in DDR RP 7 and 17-19 to avoid them This is discussed in detail in Subclause D 6

Short of intra-record tree structure, which is a deferred topic, data description is located in the first record of the file, the Data Descriptive Record (DDR) The DDR must contain, in its directory, one and only one instance of each tag that is in the subsequent Data Records (DRs). As described earlier, each tag in a directory is associated with one and only one field in the field area In the DDR, this field contains an appropriate data description Since each data field of the DRs is uniquely associated with one of these tags in its directory, there is a unique association of a data field with its data description The ISO/IEC 8211 tag in a directory entry is a unique identifier of both a data descriptive field in the DDR and an associated user data field in the DRs It is the key to ISO/IEC 8211 subfield processing

D.2.2 Level 1 data description

The data fields of a level one interchange file contain octet strings of user data and the data description is a simple field name. Thus at level 1 the simplest ISO/IEC 8211 data description is nothing but the field tag serving as an identifier and the most complex data description is a field name ISO/IEC 8211 "knows" nothing about the user data string except where it is and how long it is Any bit pattern is allowed and obviously all manner of application defined syntax and structure can exist in the string but it is undefined by the data description In fact, the data string can have all the characteristics of a composite ISO/IEC 8211 field and this is still unknown to the level 1 data description

This has one very useful capability, a very complex application data string can be easily transported in a level 1 ISO/IEC 8211 file Also a faulty data description need not stop the basic import process as it can be independent of the data description.

D.2.3 Level 2 and 3 data description

The data fields of level 2 and level 3 interchange files comprise subfields which require subfield data description, e g , extents, data types, data structures, and subfield identification Data type and data structure can be separable or related, data identification and data structure are likewise related, subfield extents may be fixed or variable or both Yet even this potentially complex topic can be addressed in increasing order of complexity The several facets of data description are listed below with subtopics in order of increasing complexity

- Subfield extents - separable
  - Fixed
  - Variable
    - System delimiter
    - User delimiters

- Data types - separable, many straight forward choices
  - Homogeneous fields - all subfields the same data type
    - Character representation
      - Text
      - Numeric
        - Integer
        - Real
        - Exponential
      - Logical
    - Binary representation
      - Unstructured bit field
      - Structured bit field
        - Unsigned integer
        - Signed integer
        - Real
        - Floating point
        - Complex
  - Inhomogeneous fields
    - All of the above

- Data structure - related to labeling
  - Regular structures
    - Elementary data items
    - Linear structures - i e , n-tuples
    - 2-D Structures - i.e , rectangular arrays
    - n-D Structures
  - Quasi-regular Structures
    - Repeating "row" tables
    - Concatenated regular structures
  - Irregular structures - loose associations, indefinite size
    - Ordered sets, lists
    - Unordered clusters

- Identification
  - Fields - Field name same as level 1
  - Subfields - related to structure
    - No identification (i e., labels)
    - Subscripts - related to structure
    - Labels - related to structure

Simple labels
Vector labels
Cartesian labels
Compound Cartesian labels

The above appears overwhelming but the following comments reduce it to a reasonable level:

1. On import, good software "knows" what to do
2 Extents and data types are separable and basically simple
3. Most data is no more complicated than an n-tuple which is easily described The most used variant, i e , a 2-D array, is a trivial addition
4. Most of the difficult questions of data description are questions of efficiency and of concern only for files containing large numbers of a particular data field In a large interchange there are usually no more than a few such field types

D.3 Data description constructs

The following presents the constructs of data description limiting the scope to the most useful forms which will meet needs of most users

D.3.1 Subfield extents

The choices here are quite simple the data is either fixed length or it is not If the subfield is variable length, the choice of a delimiter is easy use the system default, UT, or a printable character which is not in the data

There is only one efficiency concern use fixed length if possible, it may be desirable to place "nearly fixed length" data into a fixed length subfield

D.3.2 Data types

Most data is clearly of one type or another although numbers can sometimes be placed in alternate type subfields The following simple guidelines suffice

1. Use binary forms for large volumes of data (efficiency)
2 Data type character is always valid but may require application specification
3. Do not place composite integers (e g , date, YYYYMMDD) into integer subfields; use more subfields or type it as text.
4 Make use of format repetition rules.
5 Data types can be mixed in a field
6. The data type of homogeneous fields can be stipulated in the field controls, RP 1
7 Delimited homogeneous fields, except bit fields, do not require formats

D.3.3 Field identification

As in level 1 files, field identification is a simple text string, i e , Field Name.

D.3.4 Data structure without subfield identification

Data structure without subfield identification is most applicable to arrays containing numbers but not restricted to them Storage order and derived subscripts are the only identification of subfields The following guidelines suffice.

1 Indicate the dimensionality of the structure in the field controls, RP 0 (0 = 0-D, 1 = 1-D, 2 = multi-dimensional)

2 Dimension and extents

Zero and one dimension - nothing required, field exhaustion determines the number of data items. Field length must be correct Use this form for sets (ordered and unordered), lists, clusters and n-tuples with application identified items

Multi- Dimensional - use the numeric array descriptor Field length must (always) be correct Use for matrices, etc

3 If possible, do not use the variable dimension/extent feature.

D.3.5 Data structure with subfield identification

The ISO/IEC 8211 subfield identification is also used to determine the dimension and extents of structured arrays Only regular and quasi-regular structures can be labelled Regular structures have fixed dimension and fixed extents Quasi-regular structure (tables with repeating rows) have all but the first extent fixed, this extent is not labeled There is no à priori way to determine the fixed number of labels to supply for a varying extent The following guidelines will suffice for a lot of user data descriptions.

1. Indicate the dimensionality of the structure in the field controls, RP 0 (0 = 0-D, 1 = 1-d, 2 = multi-dimensional).

2 Elementary data items - a label is redundant and may be omitted

3 N-tuples - use a simple label for each data item

4 2-D Arrays - use a Cartesian label comprising a "row" vector label and a "column" vector label One row label for each row, one column label for each column Note that the identifying labels of the data items are the terms in the expansion of the Cartesian label, not the terms in the Cartesian label itself

5. Higher dimension arrays - these are an extension of the above which are seldom used, defer its study until you need it

6. Repeating row arrays - these have an unlabeled repeating "row", merely use a null row vector label Do not drop the "*" as it is needed to determine the dimension This form may also be used to reduce the overhead associated with a large number of n-tuples This conversion requires an "*" before the first label and a "2" in the structure code Row labels may also be provided as the first data item of a row

7 Compound regular structures - these are concatenations of regular structures, concatenate their labels (Cartesian, vector or simple) in the same order with "\"s as delimiters You can still have a null row vector but only one of them, in the last structure The most common concatenated structure is an n-tuple plus a 2-D structure

D.4 Large application data structures

Applications may have truly large application data structures, e g , several million n-tuples, which are unreasonably large for a single field or record. (ISO/IEC 8211 fields and records can be as large as 10^{10}-1 octets long) In this circumstance allow the application data structure to span several ISO/IEC 8211 fields or records in order that the demands on the receiver's computer will be reasonable The common tag will allow the user to re-associate the n-tuples as needed Any modern computer, even "small" ones should be able to process fields and records of 32 000 octets Above this value, inefficiencies may occur

There are large extremely regular data sets, e g , uncompressed, non-run-encoded raster data, for which there is little justification for converting to ISO/IEC 8211 even though it can be easily done and the adjoint information describing such a file is a proper candidate

D.5 Intra-record tree structures

ISO/IEC 8211 having almost no constraints on level 1 and 2 tag and field orders, has much less need to use tree structures as simple structure can be obtained by ordering It is also true that as more data originates from relational environments, there is less need Further, for instances of trees which are fixed in structure, an n-tuple can replace the tree Lastly remembering that the network, tree and relational data models are logical equivalents The receiver's point of view will be determined by the nature of their software

Having given several caveats, it is true that many application problems do have hierarchical or ordered, rooted tree structures innately connected with them For these the tree structure is quite appropriate and is not to be avoided To create an ISO/IEC 8211 tree structure, it is only necessary to do two simple things for most trees·

1) Supply the list of tag pairs in the 0...0 field after the file title

2) Order the tag and fields of both the DDR and DR in the same order as the preorder traversal sequence

For trees in which tags and fields are their own offspring, it is necessary to use the left- and right-hand branches to avoid ambiguities This is discussed in more detail in annex F

D.6 Coded character set extensions

If at present you are not sending or receiving extended character sets there will be no requirement for this facility in ISO/IEC 8211 You can ignore it if you put SPACEs in DDR RP 7 and RP 17-19

Extended character sets have little to do with the export or import of user data and a lot to do with how it is interpreted, especially, how it is displayed A fully formed user data string containing extended sets can be placed in a field by the sender and taken out again by the receiver and this should be transparent to ISO/IEC 8211 software

If you are using extended character sets, you probably know enough about ESC sequences and HOP sequences to use the ISO/IEC 8211 facilities after a little study If not, you are referred to ISO 2022 and ISO/IEC 10646 This topic is beyond the scope of this annex

Annex E
(informative)

Examples of Data Description

This annex contains several examples of data descriptive fields and the corresponding data fields An ISO/IEC 8211 file can be read by humans only with great difficulty These examples are presented as ISO/IEC 8211 fields, extracted from the ISO/IEC 8211 file but still associated with their identifying tag and octet count Each tagged field is depicted in the following input record format·

tag octet_count field_string

where the number of SPACEs between record elements is immaterial but the number of SPACEs in the data is critical to the octet count The field terminators which must occur at the end of each field are not shown

Throughout the examples, the true system delimiters, UT and FT, are unprintable and the printable characters, specified in the data descriptions, RP 4, 5, have been substituted

, - Field terminator character,
& - Subfield delimiter character

The examples are illustrated by DDF records and, if restored to a DDF, the collected data descriptions and data would form a legitimate file An example of the DDR leader and external file title field showing values in the applicable subfields is included The comments included in the data are a part of the explanation of the examples.

The data description for fields at level 2 consists of:

1) Six bytes of controls (nine if a fieldwise escape sequence is present):

a) RP - 1 data structure code
b) RP - 2 data type code
c) RP - 3,4 ancillary data type parameters for bit field
d) RP - 5 printing graphic for the field terminator, FT
e) RP - 6 printing graphic for the unit terminator, UT

2) An optional field name terminated by UT

3) An optional vector or Cartesian label terminated by UT

4) A format describing the subfield data type, subfield width or its delimiters The format is optional for many fields except for those fields containing mixed data types, fixed-width subfields, user delimiters and bit subfields

The examples presented include fixed-width subfields and variable width subfields using both standard delimiters and user-selected delimiters They are described by formats where applicable.

E.1 Leader and file title field

This example shows the prototype of a DDR leader with the user supplied values in place and the file control field for a level 2 DDF

DDR RP 5 contains a "2" and DDR RP 10-11 contains an "06" The ∇ character represents a SPACE in the following example

```
000002L∇∇∇0600000∇∇∇1102
00      24  0000,&DDF Examples File,
```

E.2 Examples of formats

E.2.1 Elementary data fields

Elementary data fields contain a single data item of any data type The data structure and data type are given in the field controls of the data description and, except for bit fields, a format is not needed The allowed elementary data fields and their data descriptions are:

```
01    14  0000;&Recid&&,
TX    13  0000,&Text&&,
E0    18  0000,&Character&&,
E1    23  0100,&Implicit Point&&,
E2    23  0200,&Explicit Point&&;
E3    30  0300,&Scaled Explicit Point&&,
E4    34  0400,&Character Mode Bit String&&,
E5    25  0500,&Bit Field&&(B(12)),
```

NOTE - 51 The fields with tags 01 and TX occur in several of the examples without further definition.

Example data fields are

```
01     5  00001
E0    59  Each field of this record contains an elementary data item
E1     5  12345
E2     6  54 321
E3    11  0 314159E+1
E4     9  101100111
E5     2  qw
```

E.2.2 Linear structures

ISO/IEC 8211 vector data fields contain linear data structures which may be of a single data type or mixed data types. The data structure and data type (except for mixed) are given in the field controls of the data description and the format is optional except for those fields which contain mixed data types, fixed-width subfields, user selected delimiters and bit fields. In this example, each field contains fixed-width and delimited subfields and a format is used to convey the subfield description

The ISO/IEC 8211 vector data structure is used to interchange a) a mathematical numeric vector, b) a n-tuple which may have mixed data types and c) an unordered set of data items of indeterminate number. The allowed vector data types and their data descriptions are

```
V0    31  1000,&Vector/Character&&(A( ));
V1     6  1100,&Vector/Implicit Point&&(2I(4),2I,2I(#));
V2    46  1200,&Vector/Explicit Point&&(2R(5),2R,2R(@)),
V3    53  1300,&Vector/Scaled Explicit Point&&(2S(8),2S,2S($));
V4    57  1400,&Vector/Character Mode Bit String&&(2C(3),2C,2C(%));
V5    32  1500,&Vector/Bit Field&&(7B(4));
V6    54  1600,&Vector/Mixed Data Types&&(A(4),I(4),A,I,S(%),C);
```

Example data fields are

```
01     5  00002
V0    59  Each field of this record contains a vector data structure.
V1    25  432187652348765&9867#2638
V2    31  36 4357 8333 4&4 786&3 56@8.965
V3    51  123 4E+2234 5E-33 254E+1&4 556E+2&345.1E-2$34 56E-4
V4    26  1010101100&10101&1001%1111
V5     4  asdf
V6    33  TEXT1234moretext&432&1 78E-1%1101
```

E.2.3 Multi-dimensioned arrays

The array structure type is used to interchange data structures of two or more dimensions containing a single data type or mixed data types The format requirement is the same as for vector fields. Some form of dimension and extent description is required Only the unlabelled fixed- and variable-dimension/extent forms are illustrated in this example. The fixed numeric descriptors are in the DDR label field. For the variable dimension/extent array (AV), the numeric descriptor is in the data field.

Appropriate data description fields are:

```
A0    35  2000;&Array/Character&2,5,2&(A( )),
A1    40  2100,&Array/Implicit Point&2,3,2&(I(4));
A2    40  2200,&Array/Explicit Point&2,2,3&(R(3)),
A3    47  2300;&Array/Scaled Explicit Point&2,2,2&(S(6)),
A4    51  2400,&Array Character Mode Bit String&2,3,2&(C(3)),
A5    39  2500,&Array/Bit Field&3,2,2,3&(12B(4)),
A6    47  2600,&Array/Mixed Data Types&2,4,2&(A(4),I(4))
AV    51  2600,&Variable Array/Mixed Data Types&&(A(4),I(4)),
```

Example data fields are

```
01     5  00003
A0    59  Each field of this record contains an array data structure
A1    24  123456789012345614829342
A2    18  1 12 23 34 45 56 6
A3    24  0 1E+12 3E-44 4E-25 5E+3
A4    18  101010110001011110
A5     6  zxcvbn
A6    32  TEST2345CASE6543NEXT8765LINE7954
AV    38  2&4&2&TEST2345CASE6543NEXT8765LINE7954
AV    38  2&2&4&TEST2345CASE6543NEXT8765LINE7954
AV    40  3&2&2&2&TEST2345CASE6543NEXT8765LINE7954
```

E.3 Examples of bit fields

This subclause provides several examples of bit subfields including variable-length bit subfields Adjacent, fixed-width bit subfields are contiguous without intervening zerofill The last subfield has zerofill to complete the octet The examples have been chosen so the bit fields plus any zerofill are ASCII characters and can be printed.

Varying-length bit subfields have a one octet count specifying the length of the adjacent bit count field (in digits) followed in turn by the bitfield data itself The next subfield starts on an octet boundary ISO/IEC 8211 does not define the meaning of bit field data, only the subfield length

Appropriate data description fields are

```
B1    36  0500,&Elementary Bit Field&&(B(30));
B2    47  1500,&Vector Bitfield&&(B(10),B(3),B(6),B(1));
B3    36  1600,&Mixed Bit Field&&(B(13),A(3)),
B4    36  1600;&Mixed Bit Field&&(A(3),B(13)),
B5    39  1600,&Mixed Bit Field&&(3(B(13),A(3))),
B6    39  1600,&Mixed Bit Field&&(3(B(13),A(#))),
B7    40  1500,&Two Varying Bit Subfields&&(2(B));
B8    47  1600,&Varying Mixed Bit Field&Bits!CHars&(B,A),
```

Example data fields are:

```
01     5  00011
TX    37  These Records Exemplify Bit Subfields
TX    58  These Fields Contain Several Combinations Of Bit Fields In
TX    57  Combination With Other Types  These Bit Fields Have Been
TX    52  Chosen To Enable Display And Are Input As Lower Case
TX    54  Characters  Varying Length Bit Subfields Are Preceded
TX    44  By A Bit Count As Delimiters Cannot Be Used
<End of DDF record>
01     5  00012
B1     4  qwer
B2     3  asd
B3     5  zxCHR
B4     5  CHRcv
B5    15  ui123JK456nm789
B6    19  ad@123#cg45#mt6574#
B7    10  19qw217asd
B8    12  215fgQWERTYz
```

E.4 Examples of binary forms

The following are examples of data descriptions for some binary forms

Appropriate data description fields are

```
b1    31  0512,&Elementary Binary Form&&,              (2 octet, unsigned integer)
b2    28  1524,&Vector Binary Form&&,                  (4 octet, signed integer)
b3    43  1600,&Mixed Binary Forms&(B22,B24,B34,B48),
b4    32  2544,Array/Real Floating&2,3,4&,
b5    49  2600,&Array/Mixed Real Floating&2,4,3&(2B44,B48),
b6    56  1600,&Vector/Mixed Data Types&integer1!words&(B11,A(#)),
```

The applicable data fields follow using printable, lower case characters to simulate binary patterns and suggest subfield contents

```
b1     2  ui
b2    16  sin1sin2sin3sin4
b3     8  sisintrlfxrealfltg
b4    48  r1c1r1c2r1c3r1c4r2c1r2c2r2c3r2c4r3c1r3c2r3c3r3c4
b5    64  r1c1r1c2r1c3longr2c1r2c2r2c3longr3c1r3c2r3c3longr4c1r4c2r4c3long
b6    28  ione-octet, unsigned integer#
```

E.5 Examples of subfield labelling

This subclause deals with labels both as subfield identifiers and array dimension/extent descriptors Labels are optional and can be applied only to those data structures which are of definite and fixed-extents The exception to this is the array with indefinitely repeating rows without "row" labels for which the columns (or higher dimensioned cross section) can be labelled Thus, "vectors" which represent instances of sets of varying numbers of items cannot be labeled nor can instances of arrays having variable dimensions/extents be labeled The label of the only subfield in a field containing a single elementary data item is always redundant, however it is permitted

E.5.1 Redundant elementary field label

The label of the only subfield of an elementary field is redundant if present.

Appropriate data description fields are.

```
e0    33  0000,&Character&redundant label&,
```

Example data fields are

```
01     5  00021
TX
e0    59  Each field of this record contains an elementary data item
```

E.5.2 Vector labels

The subfields of a vector structure are labelled by a vector label which consists of individual subfield labels separated by a "!" Any characters except "!", "*" and "\\" may be used but good practice would dictate graphics characters The number and sequence of subfield labels should agree with the number and sequence of the subfield data items they identify A subfield label can be null (i e , "!!")

Appropriate data description fields are

```
v0    31  1000,&Vector/Character&&(A( )),                    (There are no labels )
v1    81  1100,&Vector/Implicit Point& int 1!int 2!int 3!int 4!int 5!int 6&(2I(4),2I,2I(#)),
v6    88  1600,&Vector/Mixed Data Types& text 1!int 1!text 2!int !fp#!cmbf&(A(4),I(4),A,I,S(%),C);
```

Example data fields are:

```
01     5 00022
v0    59 Each field of this record contains a vector data structure
v1    25 43218765234&765&9867#2638
v6    33 TEXT1234moretext&432&1.78E-1%1101
```

E.5.3 Cartesian labels

The subfields of an array structure are labelled by a Cartesian label which when expanded (see 6 4 3 2 4 and 6 4 4 2) provides a unique label for each data item in the array. The Cartesian label is comprised of two or more vector labels separated by "*"s The number of vector labels provides the number of dimensions of the array structure and the number of subfield labels in each vector label provides the extent of the corresponding dimension

NOTE 52 - A vector structure is an array structure of dimension one

For a two dimensional array structure the order of vector labels is "row vector label"*"column vector label" This expands into row major order (i e , all columns of row one are adjacent, etc) and the array is stored in this order The Cartesian label, row1!row2*col1!col2, expands into the data labels· row1!col1, row1!col2; row2!col1, row2!col2

For arrays of higher dimension, vector labels are appended to the Cartesian label of the next lower order The row vector label may be null (i e , *"column vector label") and the remaining labels will be applied to each "row" (i e , cross section) and the unlabeled "rows" may repeat indefinitely If it is necessary to label an array of indefinitely repeating "rows", it can be accomplished by including the "row labels" as a data item in the "row" data

Appropriate data description fields are:

```
a0    35 2000,&Array/Character&2,5,2&(A( ));              (There are no labels )
a1    64 2100,&Array/Implicit Point&row 1!row 2!row 3*col 1!col 2&(I(4));
a6    54 2600;&Array/Mixed Data Types&*col 1!col 2&(A(4),I(4));
```

Example data fields are·

```
01     5 00023
a0    59 Each field of this record contains an array data structure.
a1    24 123456789012345628402451
a6    32 TEST2345CASE6543NEXT8765LINE7954
```

E.5.4 Concatenated data structures

These examples show concatenated data structures formed from the data fields having tags v6 and a1 in the above examples. There are two different descriptions of the same data. Field[c1] has a numeric descriptor and field[c2] has a Cartesian label Note that the unit terminator has been added to the vector data string and the last format term repeats to exhaust the array items of the data field

```
c1   105 2600,&Concatenated Vector\\Array& text 1!int 1!text 2!int 2!fp#!cmbf\\2,3,2&(A(4),I(4),A,I,S(%),C,(I(4))),

c2   129 2600;&Concatenated Vector\\Array&text 1!int 1!text 2!int 2!fp#!cmbf\\row 1!row 2!row 3*col 1!col 2&
           A(4),I(4),A,I,S(%),C,(I(4)));
```

Example data fields are:

```
c1    58 TEXT1234moretext&432&1.78E-1%1101&123456789012345614829342
c2    58 TEXT1234moretext&432&1.78E-1%1101&123456789012345614829342
```

Annex F
(informative)

DDF Hierarchical and Network Data Structures

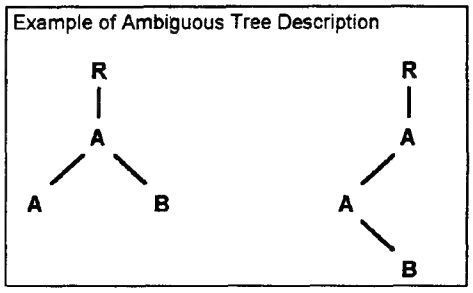
This annex contains a brief description of the conversion of some common data structures to ISO/IEC 8211 files

F.1 DDF hierarchical data structures

The hierarchical data structures of ISO/IEC 8211 are ordered, rooted trees which may be described by two methods:

1. A binary relation on the paired parent/offspring tags and the preorder traversal sequence. This method is applicable to ordered rooted trees for which the parent/offspring pairs involve different tags and has the advantage that it has almost zero overhead The data trees of the DRs must be derived from the generic tree of the DDR by the omission or repetition of the subtrees of the generic tree

NOTE 53 - The description of a tree in which a parent/offspring pair comprise the same tag may lead to ambiguities in the tree structure, i.e , different trees lead to the same preorder traversal sequence



2 The preorder traversal sequence and the left- and right-hand linkages for each node of the corresponding binary tree This method uses the position of the tags in the directory to resolve ambiguities of method 1 It has no restrictions on the parent/offspring pairs but does have additional overhead This method can describe trees which are not related to the generic tree of the DDR.

Receiving systems which support ordered, rooted trees can convert either of the above into their description method The conversion of the first method into the second method is described in F.2.

F.1.1 Forests

A forest is a collection of disjoint, ordered, rooted trees These are often described by converting them into a single ordered, rooted tree by supplying linkages from the root of each tree to the root node of the first tree This may be accomplished by either of the two above description methods. The algorithm of F 2 will parse a forest even in the absence of the supplemental links It treats the forest as an ordered set of trees based on the order of the tags in the directory

F.2 Conversion to corresponding binary tree.

This clause describes the means by which the information of Method 1 can be converted into the left- and right-hand branches of the corresponding binary tree of Method 2.

The hierarchy or ordered rooted tree structure used has data fields as its nodes and the directed logical associations between data field contents as the links in the tree structure A rooted tree has a unique node, the root, and each node may have zero or more ordered subtrees A specific instance of the tree structure for a record may have several subtrees formed from multiple occurrences of the data fields having the same tag However, such a tree must be derived from (i e , have the same logical field associations as) a generic tree for the data record which contains each field once and contains all of the possible links between fields

A rooted tree has directed links and must have only one root node which has no links entering it Each remaining node may be entered by only one link The nodes which have no departing links are called leaf nodes Once the root node is specified, the direction of all links is specified from the root node to the leaf nodes The hierarchical level is determined by numbering the root node (1) and by incrementally numbering each successive node on all paths to the leaf nodes. Trees in this document are drawn hanging from the root with the subtrees of each node ordered from left to right Examples of ordered rooted trees are shown in figure F 1 The tree described is not a binary tree which preserves left- and right-handedness in the absence of one binary branch

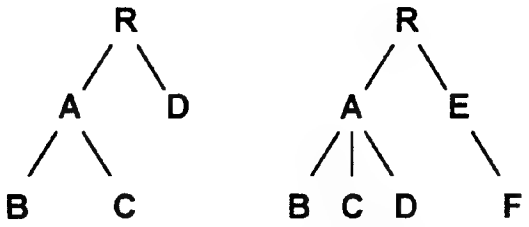


Figure F.1 - Examples of Ordered Rooted Trees

The method used here to represent rooted tree structures is closely related to the user view of the data and not the most usual computer representation of tree structures which is the corresponding binary tree (see figures F 2 and F 4).

The tags serve as node (i e , data field) identifiers. A tree structure is to be designated by two pieces of information a) the ordered pairs of node identifiers (i e , tags) of the generic structure in the top down order of parent-offspring and b) the ordered list of node identifiers (i e , tags) in the preorder traversal sequence

The generic structure of a logical record might be the tree shown in figure F 2

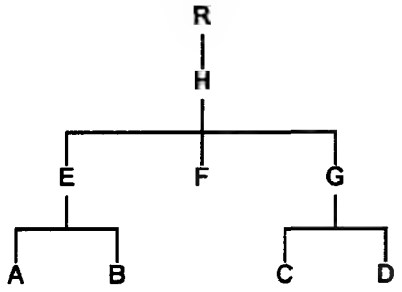


Figure F.2 - Generic Structure of a Logical Record

It has the set of ordered pairs of tags RH, HE, EA, EB, HF, HG, GC, GD, where R is the unique record identifier tag, 0...1 The preorder traversal sequence of the tags of this structure is -RHEABFGCD and each occurrence of a node is unique.

A specific instance of the previous generic structure having multiple occurrences of data fields having the same tag might be.

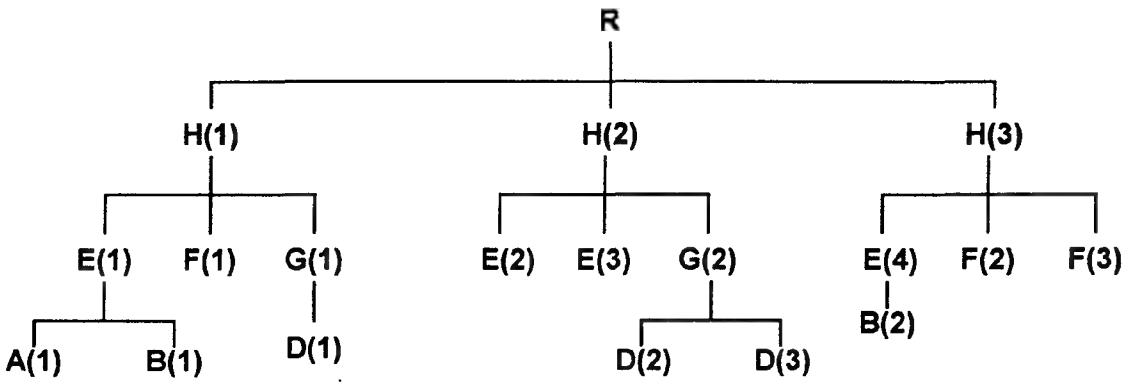


Figure F.3 - Instance of a User Data Tree based on F.2

where R is the unique record identifier tag, and ordering subscripts for repetitive tags have been added for clarity Its preorder traversal sequence is

RH(1)E(1)A(1)B(1)F(1)G(1)D(1)H(2)E(2)E(3)G(2)D(2)D(3)H(3)E(4)B(2)F(2)F(3)

which may be written without subscripts without loss of meaning Note that only the root node, R, is unique and that several instances of the field H, E, A, etc , may occur, in which case they are designated by repetition of the same tag

The preorder traversal sequence is typically combined with link information to represent the structure of an ordered rooted tree by its corresponding binary tree This binary tree has the same preorder traversal sequence as the ordered tree and the link to the first subtree of a node forms a left-hand link in the binary tree and all sibling nodes are linked in order by right-hand links between siblings These new links are easily derived from the list of tag pairs and the preorder traversal sequence. The user can then construct whatever linking convention their system requires An algorithm which produces the left- and right-hand links of the corresponding binary tree is given below using the generic tree of this annex as an example

ALGORITHM L Given that P(q) is a binary relation on a set of nodes, q, expressing the allowable directed internodal associations and T(q) is the preorder traversal sequence of a tree (or forest):

Construct L and R, vectors of indices in T, expressing the left-and right-hand linkages of the corresponding binary tree.

NOTE 54 - Read "$\notin$" as "is not an element of"
"<=" as "is set equal to", and
"|.|" as "is concatenated to"

L0 [Initialize] m <= number of pairs in P

n <= number of nodes in T
S <= 0, working stack containing the indices in T of the nodes
of a path from the root to a node, T(S(j))
L <= 0
R <= 0
L1 [Set stack to root node] j <= 1, S(1) <= 1
L2 [Set node index to root node] i <= 1
L3 [Advance to next node] i <= i + 1
L4 [Test for extent of T] if i > n end
L5 [Test paired stack node and i-th node] if T(S(j))|.|T(i) $\notin$ P go to L8
L6 [Store left link for stack node] L(S(j)) <= i
L7 [Add i-th node to stack] j <= j+1, S(j) <= i, go to L3
L8 [Test paired penultimate stack node and i-th node] if T(S(j-1))|.|T(i) $\notin$ P go to L11
L9 [Store right link for stack node] R(S(j)) <= i
L10 [Reset stack node] S(j) <= i, go to L3
L11 [Regress to lower node in path] j <= j-1,
L12 [Test for disjoint node (new tree)] if j = 0 go to L9
L13 [Retest new node pair] go to L8

Example: P: RH, HE, EA, EB, HF, GC, GD, HG

j:	1	2	3	4	5	6	6	8	9
T:	R	H	E	A	B	F	G	C	D
L:	2	3	4	0	0	0	8	0	0
R:	0	0	6	5	0	7	0	9	0

The algorithm is valid for rooted-oriented trees with or without replicate tags for nodes It will process a set of disjoint trees (i.e., a forest) The "true" branch at L12 may be set to error if it is desired to limit processing to trees.

The corresponding binary tree for the ordered rooted tree shown in figure F 2 is shown in F.4.

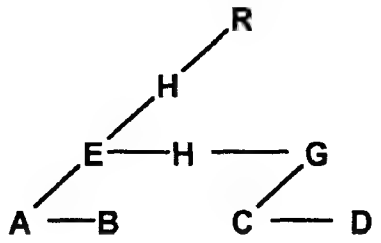


Figure F.4 - Corresponding Binary Tree to the Tree of F.2

F.3 Network data structures

Network data structures and other data structures more complex than hierarchical structures which are represented by cyclic directed graphs can be represented and transported in ISO/IEC 8211 by the use of logical or value pointers. The system dependent physical links or addresses of the host structure are removed and a logical identifier of the offspring node is placed as a value in a subfield of the parent node The direction of the pointer can be reversed if desired. This method is in common use in many systems and in those systems the value of the identifier is readily available

This method may also be used for hierarchical structures replacing the list of tag pairs and preorder traversal sequence. Relational systems use this method to permit the relational software to reconstruct the hierarchy through the appropriate queries. The reduction of a hierarchy to a set of relations permits the interchange to be accomplished as a set of n-tuples (i e , the ISO/IEC 8211 vector form) which are easily described and constructed. When this form has fixed length subfields, the overhead can be vanishingly small Conversion of the relations to the third normal form will also reduce the volume of the interchanged information. The sender may wish to consider if the receiver has relational software. ISO/IEC 8211 relational interchanges may easily be designed to automatically load into relational systems (see annex H).

Annex G
(informative)

Database Data Transfer

This annex contains information on the transfer of common data bases as ISO/IEC 8211 files.

G.1 Essential features of data base management systems

The transport of the information contained in a data base management system to a dissimilar system is a special case of file transport It differs in the very important requirement that a data base management system must have a data description of the data whereas a file does not necessarily have one The existence of the data description greatly increases the probability that the conversion of the data to an ISO/IEC 8211 transfer file can be fully automated If the data description of the data base management system can be mapped into an ISO/IEC 8211 data description representing a transfer file containing the required information in a structure that can be produced by the host system then a comprehensive conversion can be automated.

The data description of a data base management system may contain more information about a data element than that required solely for its transport This may be information on how the data element was created or how it is to be processed. In order that this portion of the data description not be lost, it is desirable to transport the original data description in its entirety either as a separate file or as a preamble to the actual user data The data structure and description for this is usually simple.

The transport of the user data can be accomplished in a format produced by a report generator or often in the "unloaded or dumped" format if the host provides one.

The recipient of such a transfer has both the ISO/IEC 8211 data description and the original data description to aid in reloading the data into a new system The original data description is of most use if the receiving system is the same as or similar to the originating system

The remainder of this annex provides guidelines for the transfer of information from three types of data base management systems relational, hierarchical and network

G.1.1 Relational data base management systems

A relational data base generally consists of several relational tables which conceptually comprise n-tuples. The relational n-tuples map directly in to ISO/IEC 8211 n-tuples Each table can be transported as a set of ISO/IEC 8211 data fields Each table can be transported as a single file If volume overhead is of consequence then the usual ISO/IEC 8211 devices for economy can be used: 1) drop the leader/directory if the regularity of the data permits or 2) place the n-tuples into repeating row arrays

The following are more specific guidelines assuming the relational system is based on SQL or a variant of SQL:

1) Transfer the SQL data description as the CREATE TABLE command in a series of ASCII text lines:

a) ISO/IEC 8211 Data Description TEXT 0000;&SQL DD&&,

or, in the notation of Annex B·

```
FIELD 'TEXT'  'SQL DD'
    A
END FIELD
```

where TEXT is the tag The corresponding SQL data description might be

```
CREATE TABLE "table name"
("column 1 name" "column 1 data type"("column 1 width"),
("column 2 name" "column 2 data type"("column 2 width"),

),
```

The descriptive information would comprise one ISO/IEC 8211 data record containing several text fields.

2) Convert the SQL CREATE TABLE command to an ISO/IEC 8211 data description

tag 1600;&field name&lbl 1!lbl 2!...&(f1(w1),f2(w2),...);

or, in the notation of Annex B

```
FIELD 'tag' 'field name'
  FOR
        'lbl1'    f1(w1)
        'lbl2'    f2(w2)
          .         ..
END FIELD
```

where

tag	is a unique ISO/IEC 8211 tag
field name	is an SQL table name
lbl i	is the SQL i-th column name
fi	is the ISO/IEC 8211 format data type code corresponding to the SQL i-th column data type,
wi	is the SQL i-th column width or an ISO/IEC 8211 delimiter if the column is variable width.

3) The user data is converted into a series of ISO/IEC 8211 fields which correspond to the data description of 2) Any of the ISO/IEC 8211 volume saving practices may be applied if desired.

If the data is to be transported as INSERT VALUES commands (this is usually much less efficient) then the ISO/IEC 8211 data description must be appropriate to these text strings

The recipient of such a transfer file must provide for the conversion to the receiving system As long as the file has been prepared in the above manner this conversion can be accomplished as the conversion of a general ISO/IEC 8211 n-tuple (including repeated row arrays) into the n-tuples of the host system The field name and subfield labels convert directly into the table name and column names and the subfield data type and extents into the column specifications The data values can be converted to INSERT VALUES commands or into the receiving system "unloaded file" With a knowledge of host variations the entire process can be automatic

G.1.2 Hierarchical data base management systems

The essential structure of hierarchical data base management systems is an ordered rooted tree whose nodes typically contain n-tuples or repeating n-tuples Since this is the same structure as an ISO/IEC 8211 file, the data usually maps easily into a transfer form with data base record types mapping into ISO/IEC 8211 fields It is necessary to convert the system dependent hierarchical links into the ISO/IEC 8211 list of tag pairs and preorder traversal sequence to preserve the tree structure As with relational data bases it is advisable to transmit the original data description in its entirety so as to provide the recipient with as much assistance as possible The steps to create a transfer file are the same as those described for the relational case except that the mappings are often more system specific due to the lack of a generally accepted hierarchical data base management standard

G.1.3 Network data base management systems

The essential structure of network data base management systems is a directed graph whose nodes typically contain n-tuples or repeating n-tuples. Since this structure is one degree of complexity above that of an ISO/IEC 8211 file, it becomes necessary to reduce the structure at least to a hierarchy preserving the host dependent network links by converting them into logical pointers and placing the pointers as data values in the appropriate nodes The user data usually maps easily into a transfer form with data base record types mapping into ISO/IEC 8211 fields. From this point the procedure follows that for hierarchies

G.2 Reduction to relational forms

The user may wish to take advantage of the fact that both hierarchical and network forms can be reduced to equivalent relational forms This is done by converting structural links to logical value links and adding them to the appropriate tables If the relational form is couched in SQL terminology, the transfer file will probably be acceptable to a wider range of receiving systems as the relational form can always be converted to the more complex form if needed and is acceptable as input to many systems. The procedure for this is merely pursuing the above procedures to their logical conclusions, i e., fields containing n-tuples.

Annex H
(informative)

Relationship to Other OSI Work

This annex contains information on the relationa of ISO/IEC 8211 to other OSI work.

ISO/IEC 8211 can be used for file transfer outside of OSI by recording the file on dismountable media for transport or within OSI by the transmission of the defined FADUs over a communication service This annex provides information which relates the ISO/IEC 8211 constructs to the constructs of the OSI Basic Reference Model and other ISO SC21 work.

H.1 OSI basic reference model

The conceptual equivalence of ISO/IEC 8211 to the Presentation layer is more easily seen if the data structure/data type concepts of ISO/IEC 8211 are rephrased in terms more common to the Presentation layer and its associated standards The following is a brief description of ISO/IEC 8211 in those terms·

" ISO/IEC 8211 - The Transfer Syntax and Encoding Rules for a Data Transfer File

" ISO/IEC 8211 is an Abstract File Transfer Syntax and associated File Transfer Semantics combined with a set of Encoding Rules for both the File Transfer Control Parameters and for the User Data Values The Abstract Syntax and Semantics for the Transfer File, Records and Fields are completely defined by the standard and any alternative encodings are announced in the Transfer Syntax Record (TSR) of the Transfer File The Transfer Syntax and Encoding Rules of the user data fields of a specific instance of a Transfer File are also present in the TSR which is the first record of the Transfer File. The remaining records are user data records the fields of which must conform to the user data syntax specified in the TSR The TSR is transferred to the receiving system and remains resident in it until file processing is complete

" The Transfer Syntax can describe a variety of Tabular structures, Sequential structures and Elementary data items The tabular forms and sequential structures have extensible options Individual data items can be fixed length or variable extent.

" The Transfer Syntax can describe a variety of Data Types and Subtypes as well as extended coded character sets The encoding rule alternatives include Basic Encoding Rules, and Packed Encoding Rules which may be intermixed. Either set of encoding rules allow further compaction when user data fields are fixed and repetitive

" While ISO/IEC 8211 specifies a Transfer File in anticipation of bulk file transport, the file is an instance of the Virtual File Store model and is thus supportable by the Sessions layer and lower layers The records of the Transfer File are hierarchical and the entire file is conceptually an ordered rooted tree, each record, a subtree, rooted to the TSR root, and each field a subtree rooted as specified in the TSR The leader/directory construct lends itself to random access within records and simple generic indices can be created to randomly access the File Store "

The above description has been phrased to emphasize the functional similarities of ISO/IEC 8211 to other Presentation layer standards especially ISO/IEC 8224 and ISO/IEC 8225 At the functional level the equivalence is accurate, at the detailed level, differences do exist as will be discussed later.

H.1.1 Other presentation layer considerations

While ISO/IEC 8211 1985 specified the syntax describing a file store it did not specify presentation layer operations on that file store it did however anticipate certain operations and, as was typical of standards of that environment, placed a brief discussion into annexes

One important consideration was that the ISO/IEC 8211 functionality would be a process (subprogram) callable by an application and used to construct an ISO/IEC 8211 file. Alternatively a standalone ISO/IEC 8211 program containing application specific subroutines could extract the necessary information from an application source to construct a transfer file At the receiving system, the reverse process with similar software was envisaged including random access methods However all of this was considered beyond the scope of an interchange standard which was to describe the file as it existed on the media

The current document does specify presentation layer operations, and makes it clearer that ISO/IEC 8211 defines a Transfer Syntax and Encoding Rules (although it uses neither ASN 1 nor its encoding rules) which can be supported by the Presentation Layer

As an example, an SQL file store may be created by a CREATE TABLE command and INSERT VALUES commands. In a like manner (albeit more complex) a CREATE ISO/IEC 8211 FILE command (containing DDR information) and a series of CREATE RECORD, CREATE FIELD commands (containing data) could be used to create ISO/IEC 8211 fields, records and files. As is appropriate to the transfer process, it is the file that is standard, not the process of producing it.

H.1.2 Remote versus local processing considerations

When viewed solely as a transfer specification for dismountable media, processes are always local unless of course the dismountable media volume is remotely mounted Whenever file transfer is viewed as a process within the OSI Basic Reference Model and especially as a session which accesses only a portion of a file store, the distinction between remote and local processes vanish

H.2 Relationship to FTAM virtual filestore model

The FTAM Virtual Filestore model can be reflected in the ISO/IEC 8211 file model and therefore it is possible to define FTAM document types which use the ISO/IEC 8211 specifications The essential FTAM Virtual Filestore structure is an ordered, rooted tree containing directories and files at its nodes This structure is augmented by references to directories and files which escalate the structure to a directed graph The structure of the files in the file store are hierarchical comprising nested File Access Data Units and Data Units

The usual description of the ISO/IEC 8211 file structure is that of a sequential file containing hierarchically structured records. It is equally accurate to describe any ISO/IEC 8211 file as an ordered rooted tree making its relationship to the FTAM files more obvious This is always valid because a sequential structure is a special case of an ordered, rooted tree as depicted below in a corresponding binary tree for an ISO/IEC 8211 file

```
        ISO_8211_file_name [host directory entry]
       /
      DDR
     /
    DR1-DR2-DR3-DR4- - - - >
```

where

the [root]	is the host directory entry for the file,
DDR	is the first (leftmost) subtree of the root, and
DRs	are siblings rooted to the DDR

At this level of detail the tree is simply very flat but nevertheless a tree as the number of DRs is not fixed and each DR may be a tree structure It should be noted that it is the varying number of subtrees that gives a structure its essential tree character; any set of fixed trees can be converted into a set of linear structures, i e , n-tuples It is also true that in a flat tree, with repeating subtrees all linked to the tree root, the list of tag pairs is known by default and the preorder traversal sequence is contained in the directory entries; it is for this reason that level 1 and 2 files are by default simple trees, only special, simpler cases of them are truly non-trees.

The DDR contains the invariant portion of the data description for each data record and when combined with the varying leader/directory (located in each DR) it forms a complete description of that DR, i e., a specific subtree This data description, in keeping with the virtual filestore model, lies at the root of each of the DR subtrees.

The occasional perception of an ISO/IEC 8211 file as being different from an FTAM file comes not from its inherent difference but from the superficial differences in their descriptions This stems in part from ISO/IEC 8211's concerns for 1) files in which there is a repetition in the DRs of the generic form of the DDR, and 2) the reduction of unnecessary overhead which is possible in such files ISO/IEC 8211 with its dropped leader/directory provides for even further overhead reduction when the DRs have a fixed format.

When viewed in this manner the equivalence of ISO/IEC 8211 files to FTAM files should be clear The same principles when applied to an ISO/IEC 8211 exchange set will also reveal the correspondence of an exchange set to a filestore.

H.2.1 Correspondence of ISO/IEC 8211 file constructs to FTAM

The following tree structure establishes the structural correspondence of the FTAM File Access Data Units (FADUs) and Data Units (DUs) to the ISO/IEC 8211 structural units

The prototypal tree for the ISO/IEC 8211 file/record structure follows.

```
FADU  ISO/IEC 8211 Data Descriptive file
  |-- FADU  ISO/IEC 8211 Logical Record
  |     |-- FADU  Record leader
  |     |-- FADU  Field Directory
  |     |     |-- FADU  Directory entry
  |     |     |
  |     |
        |-- FADU  Field area
              |-- FADU: ISO/IEC 8211 field
              |     |-- FADU  ISO/IEC 8211 Subfield
              |     |
                    |
```

The preorder traversal sequence of a tree in this form is. top down, right hand branch first

The descending "|"s indicate the existence of sibling nodes at the same level of the subtree, e g multiple logical records or multiple fields within the field area The classification of the leader and directory as FADUs is somewhat arbitrary as they contain information used to access and interpret the data area and data fields There is a one-to-one correspondence between directory entries and the fields

The first logical ISO/IEC 8211 record is the Data Description Record (DDR) which contains Data Description for each field occurring in the subsequent Data Records (DRs) For some purposes it may be desirable to root each DR to the DDR but this is not material to the equivalence of the ISO/IEC 8211 structure to the FTAM structure

In level 3 files, ISO/IEC 8211 may describe an inter-field ordered rooted tree Its preorder traversal sequence is specified, in part, by the order of the directory entries and the remaining information required is given in the DDR as the list of tag pairs. For our current purposes a level 1 or 2 file is regarded as a shallow tree.

FTAM/ISO/IEC 8211 Field Equivalencies

Since any directed structure less complex than an ordered, rooted tree can be considered as a special case of a tree, it is possible to continue the resolution of ISO/IEC 8211 fields into FTAM FADUs. The motivation for doing so depends not upon the file structure, but rather whether the "software engine" is to be considered an "FTAM engine" or an "ISO/IEC 8211 engine" In as much as either engine will make use of the DDR data descriptions the distinction seems somewhat academic Further, at the field/subfield level ISO/IEC 8211 processing becomes increasingly application oriented and thus may be considered beyond the Presentation Layer.

There is one ISO/IEC 8211 data structure, internal to fields, which deserves mention as an FTAM FADU ISO/IEC 8211 field structures having the structure code 2 are multidimensional arrays Substructures of these arrays may be regarded as FADUs For example, a 2-D array whose rows repeat can be regarded as a series of FADUs each containing a row of data

FTAM Data Units

FTAM FADUs comprise one or more Data Units (DU) which are the smallest (accessible) manipulatable data items Since ISO/IEC 8211 contains a data description detailed to the subfield level, the ISO/IEC 8211 subfield is the ultimate FTAM DU. Other aggregations of data items, e g fields and data areas, are also qualified as a DU

FTAM FADU Names

ISO/IEC 8211 equivalents to the optional FADU names are

Data area	- ISO/IEC 8211 Record Identifier
Field	- Record identifier field tag
Subfield	- Record identifier field tag subfield label (or subscripts)

H.2.2 ISO/IEC 8211 access methodology

ISO 8571 part 2 and ISO/IEC 8211 do not specify the file access methodology nor its method of implementation Both sequential and random access methodology are applicable to ISO/IEC 8211 files

Sequential Access In sequential access the FADUs comprising either the ISO/IEC 8211 records or fields are read or written sequentially

Random Access Random access of the FADUs of an ISO/IEC 8211 file requires the generation of a host system dependent access index This index may be prepared solely from the record lengths of each ISO/IEC 8211 record Random access to FADUs comprising the ISO/IEC 8211 fields is facilitated by the contents of the leader directory of each logical record Indices of this type are independent of the contents of the user data fields and could be prepared from an adjunct list of ISO/IEC 8211 record lengths

More sophisticated indices which are dependent upon data values or record/field/subfield identification can also be constructed and used to access ISO/IEC 8211 FADUs randomly However these indices are application dependent and are considered beyond the scope of an interchange standard

H.2.3 Relationship of documents to files

When a document is recorded on a medium for which volume and file structure is defined by an International Standard, then the document shall be recorded according to the provisions of that standard If more than one application standard exists, then the standard to be used shall be agreed between the parties to the interchange of the document If there is no applicable International Standard, the volume and file structure shall be agreed between the parties to the interchange of the document

The octets of the data stream representing a document shall be recorded consecutively in the data space of a file, starting at the beginning of that data space The meaning of that data space shall be as specified in the relevant standard or agreement for volume or file structure

If a document is not or cannot be recorded as one file on one volume, and the relevant standard or agreement for volume and file structure permits a file to be recorded over more than one volume, then the document shall be recorded as a multi-volume file according to the provisions of the relevant standard or agreement

If a document is not or cannot be recorded as one file on one volume, and the relevant standard or agreement for volume and file structure does not permits a file to be recorded over more than one volume, then the document shall be recorded as a number of separate files In this case the identification of the sequence of the files is subject to agreement between the parties to the interchange of the document

H.2.4 File naming

The file names of the files containing documents shall be selected by agreement between the parties to the interchange of the document This International Standard imposes no constraints on file names beyond those imposed by the requirements of the relevant standard or agreement for volume and file structure

NOTE 55 - It is suggested that where a file identifier contains an element identified as an extension field, and where permitted by the constraints on characters allowed in a file identifier, "DDF" should be used as the extension

H.3 Relationship to other syntax notations

H.3.1 Abstract syntax notation one

The comparison of ISO/IEC 8211 and ASN 1 must include the encoding rules for the latter since ISO/IEC 8211 includes its own encoding rules ASN 1 enables the user to describe the syntax of a very arbitrary information stream and apply one of several encoding rules whereas ISO/IEC 8211 anticipates an information stream comprising records and fields which in some sense, albeit complex, are repetitious and which are encoded with one encoding rule albeit composite Within their own framework they have similar functions which differ in how they are accomplished ISO/IEC 8211 gathers all the data descriptive information into a single record; ASN 1 associates the equivalent information with the data value ASN 1 uses inline tags and lengths to identify and characterize values, ISO/IEC 8211 uses disjoint tags and lengths for similar purposes ISO/IEC 8211 uses several data structures and data types, ASN 1 uses structural data types ISO/IEC 8211 envisions an extant and complete file store, ASN.1 an ephemeral information stream ISO/IEC 8211 requires that the data descriptive information accompany the data in a machine readable form; whereas ASN 1 requires that the specific instance of the data syntax exist at both ends but need not be transmitted in machine readable form ISO/IEC 8211 data description is a permanent part of a file store; parts of the ASN 1 encoded stream may not be permanent. An ISO/IEC 8211 file store lends

itself to random retrieval, random retrieval from an ephemeral data stream has little meaning and poses great technical problems.

In summary, ISO/IEC 8211 deals with an extant file store which exists or can exist within the sender's system, on a dismountable transfer medium volume or in the receiver's system When transferred by electronic means the complete file store may not be extant in the transfer system A frequent question is whether or not an ISO/IEC 8211 file can be described by ASN 1 The answer is quite probably "yes" If the question is should an ISO/IEC 8211 file be described by ASN.1, the answer is quite probably "no" The objectives of the methods are different and why would one impose an additional syntax description and encoding upon an established one The proper question is should a user with a collection of information to transfer use ASN 1 or ISO/IEC 8211? This answer is strongly dependent upon the user's objectives over and above just moving the information

H.3.2 Transfer Syntax Description Notation

The similarities and differences between ISO/IEC 8211 and this standard (TSDN) are fairly clear.

The similarities are that both are Transfer Syntaxes for files and have a lot of the same functionality

The differences are ISO/IEC 8211's objective was to totally automate the transport process and the completely machine processable Transfer Syntax must accompany the Transfer File Other goals were to allow validation of both the data description and data stream While ISO/IEC 8211 can describe a wide variety of existing data records as fields it assumes that at least the existing records will be inserted into the ISO/IEC 8211 infrastructure and perhaps some editing will be required TSDN's objective is to describe existing user files and the TSD is optionally a part of the file but may be transported independent of it possibly on hard copy This makes it easier for the sender to skip the validation step and for the receiver to experience errors Put simply, the differences are those of the new file/old file concept. Differences in the scope of data types are indicated in H 6

H.4 Relationship to data base management models

ISO/IEC 8211 can serve as a tool for the transport of both the data base description and data base content between dissimilar data base management systems The techniques for this have been described by Brooks, et al.[1] and Gallagher and Salazar[2]

Simplistically, the data base description is transported as a small simple ASCII text file It provides to the receiver, the complete description in the originating system The data contents are transported in an ad hoc form often of the nature of data base management system "dump" form Since ISO/IEC 8211 naturally accepts hierarchical and relational forms, these systems are usually amenable to automated general purpose interfaces for both sender and receiver Network forms must have host dependent links converted to logical value links usually in an n-tuple or relational form

H.5 Bibliography

The following is a list of references for further reading.

ISO 1001 1986, *Information processing - File structure and labelling magnetic tape for information interchange*

ISO 2709 1981, *Documentation - Format for bibliographic information interchange on magnetic tape*

ISO 4341 1978, *Information processing - Magnetic tape cassette and cartridge labelling and file structure for information interchange*

ISO/IEC 8632-3 1992, *Information technology - Computer graphics - Metafile for the storage and transfer of picture description information - Part 3 Binary encoding*

ISO 9293 1987, *Information processing - Volume and file structure of flexible disk cartridges for information interchange*

ISO 9660 1988, *Information processing - Volume and file structure of CD-ROM for information interchange*

---

[1] A A Brooks, H D Hammerling and B N McNeely, *User's Guide for an IBM PL/I Implementation of ISO DIS 8211 Information Processing - Specification for a Data Descriptive File for Information Interchange*, ORNL/CSD/TM-207, Oak Ridge National Laboratory, Oak Ridge TN (October 1983).

[2] L Gallagher and S Salazar, *Report on Approaches to Database Translation*, NBS Special publication 500-115, National Bureau of Standards, Gaithersburg, MD, (May 1984)

ISO/IEC 11404.—[3], *Information technology -Specification for a set of common language-independent datatypes*

D. E. Knuth, *The Art of Computer Programming, 2nd Edition Volume 1 / Fundamental Algorithms*, Chapter 2 3 - Trees, Addison_Westly Publishing Co., London, 1973

H.6 Summary of data types in other projects

The following is a collection of the principal elemental data types in use or under discussion in standards projects. This table was constructed from a survey during the preliminary studies of the requirements of this document

Data Types (1)	ISO/IEC 8211(2)	CGM (3)	CLI-IO (4)	CLI-DT (5)	TSDN (6)	ASN 1 (7)	FTN (8)
CHARACTER TYPES							
Character	A		x	x	A	29	A
Text String	A	x	e		A	29	A
Integer 6093 NR1	I	e		x	I	14	I
Real 6093 NR2	R	e			D	16	F
Real 6093 NR3	S	x			E	16	E,D
Logical/Boolean	C	x			C	13	L
Hexadecimal	A				H		
BINARY TYPES							
Bit string undefined	B	x		x	B	17,18	
Logical/Boolean	B,B1w		x	x	L		
Integer 8w bit unsigned	B1w	x					
Integer 8w bit 2-comp	B2w	x	x				
Integer 32 bit 2-comp	B24	x	x				
Real 32/64 bit IEC 559	B3w	x				16	
Real 32 bit IEC 559	B34	x	x			16	
Float 32/64 bit IEC 559	B4w	x	x				
Float 32 bit IEC 559	B44	x	x		F		
Real 32 bit	B44				R		
Double 64 bit IEC 559	B48	x					
Complex	B5w		x	x			F,E,D
Double Complex	B5w		x				D
MISCELLANEOUS							
Ordinal	I,B1w			x			
Rational	R,S,B3/4			x			
Real	R,S,B3/4			x			
Scaled	S,B4w			x			
Null	X			x			
Undefined	B			x		19	
COMPOSITE							
Date/time	A			x			

Legend

The presence of an "x" or other character in a column indicates the existence of a data type for that source (see note [2]). An "e" implies an equivalent form A character other than "x" or "e" is the data type encoding character *w* is the ISO/IEC 8211 precision. The ASN 1 entries are references to ISO/IEC 8224 clause numbers

Notes·

(1) These data descriptions are not mutually exclusive in order to introduce the terminology of the several sources
(2) The ISO/IEC 8211 form is a suggested equivalent to the form of other sources Character strings embody the conventions of ISO/IEC 646, 2022 and 10646 Complex numbers can be represented as a real couplet. Hexadecimal, which is not a standard form, can be represented as a numeric form or as characters
(3) ISO/IEC 8632-3, Computer Graphics
(4) X3T2/90-193, Computer Language Independent I/O
(5) CD/ISO/IEC 11404, Computer Language Independent Data Types
(6) ANSI BSR X3 208-199x, Transfer Syntax Description Notation
(7) ISO/IEC 8824/8825, Abstract Syntax Notation & Encoding Rules
(8) ANS X3.9 Fortran, These data types are representations on media; complex is represented as two reals

[3] To be published

Comments·

The primitive information types for this exercise can be considered as bit string, character string and the numeric types: integer, real number and complex number These information types define meaning as well as a range of values and admissible operations The numeric data types of the listing can be considered as a representation of the information types to a specified degree of precision in a character or binary mode and in a specified format which allows its interpretation as its intended information type

As a transfer syntax, ISO/IEC 8211 is intended to provide a minimal number of representation forms which will enable the effective transfer of a large percentage of the transfer communities' needs It does not attempt to meet all the needs, e g , many rational numbers cannot be transmitted except as an approximation ISO/IEC 8211 assumes these rare special needs will be met by application specific forms, e g , the rational number "1/3" will be represented exactly by the couplet "1" and "3" with implied division The unnecessary propagation of transfer data forms, which require support by all participants, is not in the best interests of the interchange community Given the foregoing, ISO/IEC 8211 contains effective equivalents to the majority of the standards' data types